Arch Appl Mech (2012) 82:1057–1074
DOI 10.1007/s00419-012-0617-5

SPECIAL ISSUE

Steffen Rothe · Ahmad-Wahadj Hamkar · Karsten J. Quint · Stefan Hartmann

Comparison of diagonal-implicit, linear-implicit and half-explicit Runge–Kutta methods in non-linear finite element analyses

Received: 10 February 2011 / Accepted: 4 August 2011 / Published online: 3 February 2012
© Springer-Verlag 2012

Abstract The method of vertical lines in the case of quasi-static solid mechanics applying constitutive models of evolutionary-type yields after the spatial discretization by means of finite elements a system of differential-algebraic equations. It is of substantial interest how fast, accurate, and stable such computations can be carried out. Moreover, the questions are how simple the implementation can be done and how susceptible a procedure is to programming errors. In this article, this is investigated for half-explicit Runge–Kutta methods that are applied to small and finite strain viscoelasticity. The advantage of the method is given by a non-iterative scheme on element level. Additionally, it turns out that for models where linear elasticity is one ingredient in the constitutive model, the method leads to only one required LU-decomposition at the beginning of the entire computation, and in each time step, only one back-substitution step has to be carried out. This outperforms current finite element computations. Order investigations of various integration schemes and the automatic step-size behavior are studied. This new proposal is compared to a classical Backward-Euler implementation, high-order stiffly accurate diagonally implicit Runge–Kutta, and recently proposed Rosenbrock-type methods. Advantages and disadvantages of the applied schemes are discussed.

Keywords Differential-algebraic equations · Viscoelasticity · High-order time-integration · Half-explicit Runge–Kutta methods · Non-linear finite elements

1 Introduction

Since the works of Wittekindt [48] and Fritzen [13], it is known that the application of the method of vertical lines to problems of quasi-static solid mechanics, which are described by constitutive models of evolutionary type, leads to a system of differential-algebraic equations after the spatial discretization using finite elements. This has been published in [12], where use is made of stiffly accurate diagonally implicit Runge–Kutta methods of higher order having the advantage that finite element programs, which are based on a Backward-Euler time-integration step on Gauss-point level, are easily extendable by a small expenditure to high-order time integration and to the possibility of step-size control based on local error estimation. This has been described in [8,11] as well as in a couple of articles for small and finite strain constitutive models, [18,20,22,28]. An alternative approach has been followed using BDF-methods in [10,41].

All these methods lead after the time-integration step to a system of non-linear equations (depending on the constitutive model and whether geometrical non-linearity is assumed). It turned out in [26] and [23] that linear-implicit Rosenbrock-type method lead to completely iteration-less finite element schemes that drastically reduce the computational times. This has been demonstrated for the case of smooth constitutive models in small and finite strain viscoelasticity.

S. Rothe · A.-W. Hamkar · K. J. Quint · S. Hartmann (✉)
Institute of Applied Mechanics, Clausthal University of Technology,
Adolph-Roemer-Str. 2A, 38678 Clausthal-Zellerfeld, Germany
E-mail: stefan.hartmann@tu-clausthal.de

An interesting alternative is given by half-explicit Runge–Kutta methods (HERK-methods), see [14,15]. For further reading in the context of PDAEs, see [4], where an half-explicit linear one-step method is applied. For the case of semi-explicit DAEs of index 1 combined with terminated Newton iterations, it is referred to [3], and for index 2 problems, which arise in the case of rate-independent elasto-plasticity problems, see [2,37,42]. The entire HERK-approach in this article, however, is different to a pure application of an explicit scheme on Gauss-point level to solve the local ODE or DAE of the constitutive model such as in [31]. Accordingly, special investigations are necessary.

The application of HERK-methods to the DAE-interpretation in non-linear finite elements represents a scheme, where the differential part of the DAE-system is integrated by an explicit time-integration scheme, and the algebraic part is fulfilled exactly at each stage (point in time). The latter is represented by the discretized weak formulation (equilibrium conditions). Commonly, small strain constitutive models are based on a sub-model of linear, isotropic elasticity (Hooke's law), yielding for the principle of virtual displacements a linear relation in the nodal displacements. It will be shown that in this case there is no system of non-linear equations to be solved. For the resulting linear system, which has the same coefficient matrix (stiffness matrix), only back-substitutions in all time steps must be carried out having the advantage of leading to a very fast solution scheme. This holds, for example, for the three-parameter model of linear viscoelasticity and further models, where the viscosity is process dependent.

In the case of finite deformation, the geometrical non-linear part and the underlying non-linear strain measure (non-linear in view of the displacement field) require inevitably global iterations, because the discretized weak form leads in any case to a system of non-linear equations. Thus, Newton–Raphson or Newton-like methods can be applied. However, no local iterations arise leading to a profit in efficiency. It will be shown how to apply extrapolation schemes for the starting value of the non-linear solution procedure leading to very efficient computations, particularly, for the modified Newton method.

The proposed procedures for small and finite deformations are compared to classical Backward-Euler implementations, higher-order diagonally implicit Runge–Kutta methods, where problem-adapted stress-algorithms were developed for the applied constitutive models, see for the case of small strain viscoelasticity [19] and for large strain viscoelasticity [18,23], and with Rosenbrock-type methods.

The notation in use is defined in the following manner: geometrical vectors are symbolized by $\mathbf{a}$, second-order tensors $\mathbf{A}$ by bold-faced Roman letters, and calligraphic letters $\mathcal{A}$ define fourth order tensors. Furthermore, we introduce matrices at global level symbolized by bold-faced italic letters $\boldsymbol{A}$, and at local (element) level by bold-faced sans-serif letters A.

2 Constitutive modeling

Constitutive models of evolutionary-type occur in the field of viscoelasticity, rate-independent plasticity, and viscoplasticity. Here, two types, namely a model of small strain viscoelasticity and a model for finite deformations representing rate-dependent effects, are considered.

2.1 Small strain viscoelasticity

First, a small strain viscoelasticity model is taken into consideration, where the stress state $\mathbf{T}$ decomposes into an equilibrium and an overstress part

$$\mathbf{T} = \mathbf{T}_{\mathrm{eq}} + \mathbf{T}_{\mathrm{ov}}. \tag{1}$$

Additionally, an additive decomposition of the strain tensor $\mathbf{E}$ into an elastic and a viscous part is assumed

$$\mathbf{E} = \mathbf{E}_{\mathrm{e}} + \mathbf{E}_{\mathrm{v}}. \tag{2}$$

Here, the viscous strains $\mathbf{E}_{\mathrm{v}}$ evolve according to the flow rule

$$\dot{\mathbf{E}}_{\mathrm{v}} = \frac{1}{\eta(\mathbf{E}, \mathbf{E}_{\mathrm{v}})}\mathbf{T}_{\mathrm{ov}} = \frac{2\hat{G}}{\eta(\mathbf{E}, \mathbf{E}_{\mathrm{v}})}\mathbf{E}_{\mathrm{e}}^{D} = \frac{2\hat{G}}{\eta(\mathbf{E}, \mathbf{E}_{\mathrm{v}})}(\mathbf{E} - \mathbf{E}_{\mathrm{v}})^{D}, \tag{3}$$

where the overstress is defined by

$$\mathbf{T}_{\mathrm{ov}} = 2\hat{G}\mathbf{E}_{\mathrm{e}}^{D}. \tag{4}$$

$\hat{G}$ represents the shear modulus of the overstress part and η denotes a process-dependent viscosity that is assumed to be

$$\eta(\mathbf{E}, \mathbf{E}_{\mathrm{v}}) = \eta_0 e^{-s_0 \|\mathbf{T}^D_{\mathrm{ov}}\|} = \eta_0 e^{-2\hat{G} s_0 \|(\mathbf{E}-\mathbf{E}_{\mathrm{v}})^D\|}, \tag{5}$$

see [19] and the literature cited therein.

Additionally, the equilibrium stresses $\mathbf{T}_{\mathrm{eq}}$ are assumed to represent linear elasticity,

$$\mathbf{T}_{\mathrm{eq}} = K(\operatorname{tr}\mathbf{E})\mathbf{I} + 2G\mathbf{E}^D. \tag{6}$$

K and G define the bulk and shear modulus, $\operatorname{tr}\mathbf{E} := E_{kk}$ the trace operator and $\mathbf{E}^D = \mathbf{E} - (\operatorname{tr}\mathbf{E})/3\mathbf{I}$ the deviator of a second-order tensor. Additionally, $\mathbf{I}$ symbolizes the identity tensor of second-order, $\mathbf{IA} = \mathbf{A}$.

The mathematical structure of model (1), (3)–(6) is given by

$$\mathbf{T} = \mathbf{h}(\mathbf{E}, \mathbf{E}_{\mathrm{v}}), \tag{7}$$

$$\dot{\mathbf{E}}_{\mathrm{v}} = \mathbf{r}_{\mathrm{v}}(\mathbf{E}, \mathbf{E}_{\mathrm{v}}), \tag{8}$$

or in matrix notation, where the coefficients of the symmetric stress and strain tensors are assembled in a (6×1)-column vector, $\mathbf{T}^T = \{t_{11}\, t_{22}\, t_{33}\, t_{12}\, t_{23}\, t_{31}\}$,

$$\mathbf{T} = \mathbf{CE} + 2\hat{G}\mathbf{D}\{\mathbf{E} - \mathbf{E}_{\mathrm{v}}\} = \mathbf{A}_1\mathbf{E} - \mathbf{A}_2\mathbf{E}_{\mathrm{v}}, \tag{9}$$

$$\dot{\mathbf{E}}_{\mathrm{v}} = \frac{2\hat{G}}{\eta(\mathbf{E}, \mathbf{E}_{\mathrm{v}})}\mathbf{D}\{\mathbf{E} - \mathbf{E}_{\mathrm{v}}\}, \tag{10}$$

with the constant matrices $\mathbf{A}_1 = \mathbf{C} + 2\hat{G}\mathbf{D} = \mathbf{C} + \mathbf{A}_2$, $\mathbf{A}_1 \in \mathbb{R}^{6\times6}$, and $\mathbf{A}_2 = 2\hat{G}\mathbf{D}$, $\mathbf{A}_2 \in \mathbb{R}^{6\times6}$. $\mathbf{D} \in \mathbb{R}^{6\times6}$ represents the deviator operator in matrix form, see [19]. The stress tensor $\mathbf{T}$ (or the column vector representation $\mathbf{T}$ as well) depends linearly on the strain tensor $\mathbf{E}$, that is, $\mathbf{E}$, which can be exploited in the numerical treatment (this does also not change if a couple of Maxwell-models are assembled in parallel, i.e., $\mathbf{T}_{\mathrm{ov}} = \sum_{k=1}^{n_{\mathrm{ov}}} \mathbf{T}_{\mathrm{ov}k}$, $\mathbf{T}_{\mathrm{ov}k} = 2\hat{G}_k(\mathbf{E} - \mathbf{E}_{\mathrm{v}k})^D$ having the flow-rules $\dot{\mathbf{E}}_{\mathrm{v}k} = (1/\eta_k(\mathbf{E}, \mathbf{E}_{\mathrm{v}1}, \ldots, \mathbf{E}_{\mathrm{v}n_{\mathrm{ov}}}))\mathbf{T}_{\mathrm{ov}k}$).

2.2 Finite strain viscoelasticity

The counterpart in finite strain viscoelasticity has a similar structure. However, the exact description of the deformation is based on the multiplicative decomposition of the deformation gradient, $\mathbf{F} = \hat{\mathbf{F}}_{\mathrm{e}}\hat{\mathbf{F}}_{\mathrm{v}}$ with $\mathbf{F}(\mathbf{X}, t) = \operatorname{Grad}\boldsymbol{\chi}_{\mathrm{R}}(\mathbf{X}, t)$, where $\mathbf{x} = \boldsymbol{\chi}_{\mathrm{R}}(\mathbf{X}, t)$ defines the motion of particle $\mathbf{X}$ occupying the place $\mathbf{x}$ at time t. In the following, the constitutive model proposed in [23] on the basis of the viscoelasticity model of Hartmann [18], see for the origin in the case of incompressibility [33,35], is applied. The model has the same structure of the model mentioned in Sect. 2.1. However, the stress state $\tilde{\mathbf{T}} = \tilde{\mathbf{T}}_{\mathrm{eq}} + \tilde{\mathbf{T}}_{\mathrm{ov}}$, here represented by second Piola-Kirchhoff tensors, is given by the equilibrium stress

$$\tilde{\mathbf{T}}_{\mathrm{eq}} = 2\rho_{\mathrm{R}}\frac{\mathrm{d}\overline{\psi}}{\mathrm{d}\mathbf{C}} = \rho_{\mathrm{R}}J^{1/3}U'(J)\overline{\mathbf{C}}^{-1} + 2\rho_{\mathrm{R}}J^{-2/3}\left((w_1 + w_2\mathrm{I}_{\overline{\mathbf{C}}})\mathbf{I} - w_2\overline{\mathbf{C}} - \tfrac{1}{3}(w_1\mathrm{I}_{\overline{\mathbf{C}}} + 2w_2\mathrm{II}_{\overline{\mathbf{C}}})\overline{\mathbf{C}}^{-1}\right) \tag{11}$$

with

$$U'(J) = \frac{K}{10}(J^4 - J^{-6}), \quad w_1 = c_{10} + 3\alpha\mathrm{I}^2_{\overline{\mathbf{C}}}, \quad w_2 = c_{01}\frac{3}{2}\mathrm{II}^{1/2}_{\overline{\mathbf{C}}}, \tag{12}$$

and the over-stresses

$$\tilde{\mathbf{T}}_{\mathrm{ov}} = 2\rho_{\mathrm{R}}\mu\frac{(\det\mathbf{C}_{\mathrm{v}})^{1/3}}{(\det\mathbf{C})^{1/3}}\left(\mathbf{C}_{\mathrm{v}}^{-1} - \tfrac{1}{3}(\mathbf{C}\cdot\mathbf{C}_{\mathrm{v}}^{-1})\mathbf{C}^{-1}\right), \tag{13}$$

see [18,23]. The equilibrium stress state is based on a poly-convex strain-energy function $\overline{\psi}(J, \mathrm{I}_{\overline{\mathbf{C}}}, \mathrm{II}_{\overline{\mathbf{C}}})$ proposed by Hartmann and Neff [25]. Here, $\overline{\mathbf{C}} = (\det\mathbf{C})^{-1/3}\mathbf{C}$ represents the unimodular and $\mathbf{C}$ the classical right Cauchy–Green tensors. $J = \det\mathbf{F}$ symbolizes the determinant of the deformation gradient and ρ_{R} defines the

density in the reference configuration. $\mathbf{C}_\mathrm{v} = \hat{\mathbf{F}}_\mathrm{v}^T \hat{\mathbf{F}}_\mathrm{v}$ denotes the viscous right Cauchy–Green tensor that evolves according to

$$\dot{\mathbf{C}}_\mathrm{v} = \tilde{\mathbf{r}}(\mathbf{C}, \mathbf{C}_\mathrm{v}) = \frac{4\rho_\mathrm{R}\mu}{\eta} \frac{(\det \mathbf{C}_\mathrm{v})^{1/3}}{(\det \mathbf{C})^{1/3}} \left(\mathbf{C} - \tfrac{1}{3}(\mathbf{C} \cdot \mathbf{C}_\mathrm{v}^{-1})\mathbf{C}_\mathrm{v}\right). \tag{14}$$

$$\eta(\mathbf{C}, \mathbf{C}_\mathrm{v}) = \eta_0 \exp\left(-s_0 \sqrt{\mathbf{C}\tilde{\mathbf{T}}_\mathrm{ov} \cdot \tilde{\mathbf{T}}_\mathrm{ov}\mathbf{C}}\right) \tag{15}$$

is the process-dependent viscosity. The material parameters of the equilibrium stress part are α, c_{01}, c_{10}, and K and of the overstress part η_0, s_0 and μ. In conclusion, the constitutive model has the structure

$$\tilde{\mathbf{T}} = \tilde{\mathbf{h}}(\mathbf{C}, \mathbf{C}_\mathrm{v}), \tag{16}$$

$$\dot{\mathbf{C}}_\mathrm{v} = \tilde{\mathbf{r}}(\mathbf{C}, \mathbf{C}_\mathrm{v}) \tag{17}$$

and fits into a wide range of rate-type models.

3 Space and time discretization

Before discussing half-explicit Runge–Kutta methods and their effects on non-linear finite elements, the basic DAE-system has to be recapped. After introducing HERK-methods, formerly applied DIRK methods and Rosenbrock-type methods are shortly repeated and their implementations are discussed.

3.1 Space discretization: small strain case

The spatial discretization of the local balance of linear momentum in the context of finite elements

$$\operatorname{div} \mathbf{T} + \rho\mathbf{k} = \mathbf{0}$$

makes use of the weak formulation (principle of virtual displacements)

$$\int_V \mathbf{T} \cdot \delta\mathbf{E} \, \mathrm{d}V = \int_A \mathbf{t} \cdot \delta\mathbf{u} \, \mathrm{d}A + \int_V \rho\mathbf{k} \cdot \delta\mathbf{u} \, \mathrm{d}V,$$

where V defines the volume and A the surface of the material body, $\delta\mathbf{E} = 1/2(\operatorname{grad} \delta\mathbf{u} + \operatorname{grad}^T \delta\mathbf{u})$ is the virtual strain tensor, $\delta\mathbf{u}$ the virtual displacements, $\mathbf{t} = \mathbf{T}\mathbf{n}$ the surface tractions, and $\rho\mathbf{k}$ the specific body forces. The application of the Galerkin method of finite elements and the Gauss-integration yields the system of algebraic equations

$$\boldsymbol{g}(t, \boldsymbol{u}, \boldsymbol{q}) = 0 \tag{18}$$

with the unknown nodal displacements $\boldsymbol{u} \in \mathbb{R}^{n_\mathrm{u}}$ and the time t. The explicit time-dependence results from the externally prescribed equivalent nodal forces $\overline{\boldsymbol{p}}(t) \in \mathbb{R}^{n_\mathrm{u}}$ (Neumann boundary conditions) and nodal displacements $\overline{\boldsymbol{u}}(t) \in \mathbb{R}^{n_\mathrm{p}}$ (Dirichlet boundary conditions). The vector $\boldsymbol{q} \in \mathbb{R}^{n_\mathrm{Q}}$ represents all internal variables of all Gauss-points of the discretized structure. In more detail, Eq. (18) is defined by

$$\boldsymbol{g}(t, \boldsymbol{u}, \boldsymbol{q}) := \sum_{e=1}^{n_\mathrm{e}} \boldsymbol{Z}^{eT} \left\{ \sum_{j,k,l} w_{jkl} \mathbf{B}^{e(jkl)^T} \, \mathbf{h}\left(\mathbf{E}^{e(jkl)}(t), \mathbf{q}^{e(jkl)}(t)\right) \det \mathbf{J}^{e(jkl)} \right\} - \overline{\boldsymbol{p}}(t), \tag{19}$$

where $\boldsymbol{Z}^e \in \mathbb{R}^{3n_\mathrm{en} \times n_\mathrm{u}}$ symbolizes the assembly matrix in the 3D-case (which is not programmed, but reflecting the assemblage of all element quantities). $\boldsymbol{Z}^e$ and $\overline{\boldsymbol{Z}}^e \in \mathbb{R}^{3n_\mathrm{en} \times n_\mathrm{p}}$ assign the global unknown and known nodal displacements $\boldsymbol{u} \in \mathbb{R}^{n_\mathrm{u}}$ and $\overline{\boldsymbol{u}} \in \mathbb{R}^{n_\mathrm{p}}$ to the element nodal displacements

$$\mathbf{u}^e = \boldsymbol{Z}^e \boldsymbol{u} + \overline{\boldsymbol{Z}}^e \overline{\boldsymbol{u}}, \tag{20}$$

where n_{en} is the number of element nodes. For an eight-noded hexahedral element $n_{\mathrm{en}} = 8$ holds, $\mathbf{u}^e \in \mathbb{R}^{3n_{\mathrm{en}}}$. $w_{jkl} := w_j^{\mathrm{G}} w_k^{\mathrm{G}} w_l^{\mathrm{G}}$ represents the product of the weighting factors of the Gauss-integration (or other integration formulas) in the ξ, η and ζ direction, $\mathbf{B}^{e(jkl)} \in \mathbb{R}^{6\times 3n_{\mathrm{en}}}$ denotes the strain-displacement matrix in element e at Gauss-point $\boldsymbol{\xi}^{jkl} = \boldsymbol{\varphi}^e(\mathbf{x}) = \{\xi_j, \eta_k, \zeta_l\}$ defining the strain vector

$$\mathbf{E}^{e(jkl)} = \mathbf{B}^{e(jkl)}\mathbf{u}^e = \mathbf{B}^{e(jkl)}\boldsymbol{Z}^e\boldsymbol{u} + \mathbf{B}^{e(jkl)}\overline{\boldsymbol{Z}}^e\overline{\boldsymbol{u}}. \tag{21}$$

$\mathbf{J}^{e(jkl)} := \mathbf{J}^e(\boldsymbol{\xi}^{e(jkl)}) = \partial\boldsymbol{\chi}^e/\partial\boldsymbol{\xi}|_{\boldsymbol{\xi}=\boldsymbol{\xi}^{jkl}}$ with $\boldsymbol{\varphi}^e = \boldsymbol{\chi}^{e-1}$ is the Jacobian of the coordinate transformation from real to element coordinates, $\mathbf{x} = \boldsymbol{\chi}^e(\boldsymbol{\xi})$. The symbol $\sum_{j,k,l} := \sum_{j=1}^{n_\xi}\sum_{k=1}^{n_\eta}\sum_{l=1}^{n_\zeta}$ abbreviates the triple sum over the Gauss-points in an element in the three local coordinate directions, see [19,29].

$\mathbf{q}^{e(jkl)} \in \mathbb{R}^{n_{\mathrm{q}}}$ is the vector of the internal variables in element e at Gauss-point $\boldsymbol{\xi}^{jkl}$, see models (3) or (14), where $n_{\mathrm{q}} = 6$ holds for both models. Totally, $n_{\mathrm{Q}} = n_{\mathrm{e}} \times n_{\mathrm{i}} \times n_{\mathrm{q}}$ occur in the entire structure, with n_{e} elements and n_{i} integration points of all elements. The assemblage of the internal variables and the differential equations

$$\boldsymbol{q}(t) = \sum_{e,j,k,l} \boldsymbol{Z}_{\mathsf{q}}^{e(jkl)^T}\mathbf{q}^{e(jkl)}(t), \quad \boldsymbol{r}(t, \boldsymbol{u}(t), \boldsymbol{q}(t)) = \sum_{e,j,k,l} \boldsymbol{Z}_{\mathsf{q}}^{e(jkl)^T}\mathbf{r}(\mathbf{E}^{e(jkl)}(t), \mathbf{q}^{e(jkl)}(t)) \tag{22}$$

leads to the ODE-system (differential part of the DAE-system)

$$\dot{\boldsymbol{q}}(t) - \boldsymbol{r}(t, \boldsymbol{u}(t), \boldsymbol{q}(t)) = \boldsymbol{0}, \quad \boldsymbol{q}(t_0) = \boldsymbol{q}_0. \tag{23}$$

The matrices $\boldsymbol{Z}_{\mathsf{q}}^{e(jkl)} \in \mathbb{R}^{n_{\mathrm{q}}\times n_{\mathrm{Q}}}$ represent the compilation process (incidences) of the internal variables of each Gauss-point into a the large vector $\boldsymbol{q} \in \mathbb{R}^{n_{\mathrm{Q}}}$, or vice versa at Gauss-point $\boldsymbol{\xi}^{jkl}$ the internal variables are given by $\mathbf{q}^{e(jkl)}(t) = \boldsymbol{Z}_{\mathsf{q}}^{e(jkl)}\boldsymbol{q}(t)$. The $\boldsymbol{Z}_{\mathsf{q}}^{e(jkl)}$-matrices are not explicitly programmed. They stand for the assemblage procedure. Accordingly, the coupled system (18) and (23) represents a system of differential-algebraic equations, which has to solved by an appropriate time-integration method.

3.2 Half-explicit Runge–Kutta methods

In this article, use is made of half-explicit Runge–Kutta methods, see [14,15,47]. The stage computation applied to the DAE-system (18) and (23) reads

$$\boldsymbol{Q}_{ni} = \boldsymbol{q}_n + \Delta t_n \sum_{j=1}^{i-1} a_{ij}\boldsymbol{r}(T_{nj}, \boldsymbol{U}_{nj}, \boldsymbol{Q}_{nj}) \tag{24}$$

$$\boldsymbol{g}(T_{ni}, \boldsymbol{U}_{ni}, \boldsymbol{Q}_{ni}) = \boldsymbol{0} \quad \rightsquigarrow \quad \boldsymbol{U}_{ni} \tag{25}$$

for $i = 1, \ldots, s$ with the step-size $\Delta t_n = t_{n+1} - t_n$, the stage time $T_{ni} = t_n + c_i\Delta t_n$, the weighting factors of the underlying explicit Runge–Kutta method, a_{ij} with $a_{ij} = 0$ for $j \geq i$, and $c_i = \sum_{j=1}^{s} a_{ij}$, $i = 1, \ldots, s$. s is the number of stages.

$$\dot{\boldsymbol{Q}}_{ni} = \boldsymbol{r}(T_{ni}, \boldsymbol{U}_{ni}, \boldsymbol{Q}_{ni}) \tag{26}$$

represents the stage derivative of the internal variables. In other words, (24)–(25) have to be computed $s - 1$ times (the first step, $\boldsymbol{Q}_{\mathrm{n1}} = \boldsymbol{q}_n$, $\boldsymbol{U}_{\mathrm{n1}} = \boldsymbol{u}_n$, $T_{\mathrm{n1}} = t_n$, is always fulfilled, see Eqs. (27)–(28)). Obviously, $\boldsymbol{Q}_{ni}$ in Eq. (24) depends only on previously computed quantities and the equilibrium condition (25) on the unknown nodal displacements $\boldsymbol{U}_{ni}$ at stage T_{ni} because $\boldsymbol{Q}_{ni}$ is known from step (24). The final quantities at time t_{n+1} are computed by

$$\boldsymbol{q}_{n+1} = \boldsymbol{q}_n + \Delta t_n \sum_{i=1}^{s} b_i\dot{\boldsymbol{Q}}_{ni} \tag{27}$$

$$\boldsymbol{g}(t_{n+1}, \boldsymbol{u}_{n+1}, \boldsymbol{q}_{n+1}) = \boldsymbol{0} \quad \rightsquigarrow \quad \boldsymbol{u}_{n+1} \tag{28}$$

Equations (24)–(27) are explicit, that is, only function evaluations have to be carried out. Caused by the formal assemblage (22), the computations of the internal variables can be performed on Gauss-point level. The

$$
\begin{array}{c|cccc}
c_1 & 0 & 0 & \dots & 0 \\
c_2 & a_{21} & 0 & \ddots & \vdots \\
\vdots & \vdots & & \ddots & 0 \\
c_s & a_{s1} & a_{s2} & \dots & 0 \\
\hline
 & b_1 & b_2 & \dots & b_s \\
 & \hat{b}_1 & \hat{b}_2 & \dots & \hat{b}_s
\end{array}
\qquad
\begin{array}{c|cccc}
c_1 & 0 & 0 & \dots & 0 \\
c_2 & a_{21} & 0 & \ddots & \vdots \\
\vdots & \vdots & & \ddots & 0 \\
c_s & b_1 & b_2 & \dots & 0 \\
\hline
 & b_1 & b_2 & \dots & 0 \\
 & \hat{b}_1 & \hat{b}_2 & \dots & \hat{b}_s
\end{array}
\qquad
\begin{array}{c|cccc}
c_1 & a_{11} & 0 & \dots & 0 \\
c_2 & a_{21} & a_{22} & \ddots & \vdots \\
\vdots & \vdots & & \ddots & 0 \\
c_s & b_1 & b_2 & \dots & b_s \\
\hline
 & b_1 & b_2 & \dots & b_s \\
 & \hat{b}_1 & \hat{b}_2 & \dots & \hat{b}_s
\end{array}
$$

(a) **(b)** **(c)**

Fig. 1 Butcher array for various Runge–Kutta methods. **a** Explicit Runge–Kutta scheme. **b** Stiffly accurate, explicit Runge–Kutta scheme. **c** Stiffly accurate, diagonally implicit Runge–Kutta scheme

coefficients c_i, a_{ij}, and b_i of the method are assembled in Fig. 1a. Currently, s systems of non-linear equations have to be calculated (($s-1$-times Eq. (25) and one time Eq. (28)). This is minimized by applying a stiffly accurate method, where the coefficients $b_j = a_{sj}$, $j = 1, \dots, s$, are chosen, see Fig. 1b. Accordingly, the final step (27)–(28) can be omitted so that only $s-1$ systems of non-linear equations have to solved.

The application of this procedure to small strain viscoelasticity (9)–(10) with the process-dependent viscosity (5) yields the equilibrium equation

$$\boldsymbol{g}(t, \boldsymbol{u}, \boldsymbol{q}) := \boldsymbol{K}\boldsymbol{u} + \overline{\boldsymbol{K}}\,\overline{\boldsymbol{u}}(t) - \boldsymbol{K}_{\mathrm{q}}\boldsymbol{q} - \overline{\boldsymbol{p}}(t) = \boldsymbol{0} \tag{29}$$

which is linear in the unknown nodal displacements $\boldsymbol{u}$ and the internal variables $\boldsymbol{Q}$. Here, the indices ni of the stage or the final value $n+1$ are omitted for brevity. The abbreviations in Eq. (29) are

$$\boldsymbol{K} = \sum_{e=1}^{n_{\mathrm{e}}} \boldsymbol{Z}^{eT} \left[\sum_{j,k,l} w_{jkl} \mathbf{B}^{e(ijk)^T} \mathbf{A}_1 \mathbf{B}^{e(ijk)} \det \mathbf{J}^{e(ijk)} \right] \boldsymbol{Z}^e, \tag{30}$$

$$\overline{\boldsymbol{K}} = \sum_{e=1}^{n_{\mathrm{e}}} \boldsymbol{Z}^{eT} \left[\sum_{j,k,l} w_{jkl} \mathbf{B}^{e(ijk)^T} \mathbf{A}_1 \mathbf{B}^{e(ijk)} \det \mathbf{J}^{e(ijk)} \right] \overline{\boldsymbol{Z}}^e, \tag{31}$$

$$\boldsymbol{K}_{\mathrm{q}} = \sum_{e=1}^{n_{\mathrm{e}}} \boldsymbol{Z}^{eT} \left[\sum_{j,k,l} w_{jkl} \mathbf{B}^{e(ijk)^T} \mathbf{A}_2 \det \mathbf{J}^{e(ijk)} \right] \boldsymbol{Z}_{\mathrm{q}}^{e(jkl)}. \tag{32}$$

In brackets of Eq. (30) or (31), the element stiffness matrix is visible. In other words, the linear system

$$\boldsymbol{K}\boldsymbol{u} = \overline{\boldsymbol{p}}(t) + \boldsymbol{K}_{\mathrm{q}}\boldsymbol{q} - \overline{\boldsymbol{K}}\,\overline{\boldsymbol{u}}(t) \tag{33}$$

with a constant coefficient (stiffness) matrix $\boldsymbol{K}$ for all times t has to be solved, that is, at time $t_0 = 0$, one LU-decomposition can be performed, and for all further times t_{n+1} (and T_{ni}), only back-substitutions of Eq. (33) are necessary. Only if a non-linear elasticity relation occurs, $\mathbf{T} = \mathbf{h}(\mathbf{E}, \mathbf{q})$, the Newton–Raphson method has to be used, see [19] for details within finite elements, leading in each iteration-step (m) at stage T_{ni} or the final step t_{n+1} to the linear system

$$\left[\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}} \right] \Bigg|_{\boldsymbol{u}=\boldsymbol{U}_{ni}^{(m)}} \Delta \boldsymbol{U} = -\boldsymbol{g}(T_{ni}, \boldsymbol{U}_{ni}^{(m)}, \boldsymbol{Q}_{ni}) \tag{34}$$

with $\Delta \boldsymbol{U} = \boldsymbol{U}_{ni}^{(m+1)} - \boldsymbol{U}_{ni}^{(m)}$. The advantage in HERK-methods lies in the fact that the consistent tangent operator resulting from the implicit function theorem in the Multilevel-Newton algorithm (MLNA), which is usually applied in finite elements, see [19], is not required (only for the elasticity relation). More precisely, for the small strain problem (19), one needs merely

$$\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}} = \sum_{e=1}^{n_{\mathrm{e}}} \boldsymbol{Z}^{eT} \mathbf{k}^e \boldsymbol{Z}^e \tag{35}$$

with the (tangential) element stiffness matrix

$$\mathbf{k}^e = \sum_{j,k,l} w_{jkl} \mathbf{B}^{e(jkl)^T} \left. \frac{\partial \mathbf{h}}{\partial \mathbf{E}} \right|_{\mathbf{E}=\mathbf{E}^{e(jkl)}} \mathbf{B}^{e(jkl)} \det \mathbf{J}^{e(jkl)}, \tag{36}$$

that is, the elasticity relation (7) has to be linearized with respect to the strains (application of the chain-rule in view of Eq. (21).

In the case of finite deformations, it is common to apply mixed finite element formulations to reduce or circumvent volumetric locking phenomena. These problems result from the nearly incompressible formulation of the constitutive model. Here, use is made of a three-field mixed variational equations, where the displacements, the pressure, and the volumetric deformation are discretized, see [18,34,44,45]. The basic property of this approach is the preceding elimination of the pressure and volumetric degrees of freedom on element level by a Newton–Schur step (these unknowns are linearly embedded in the equation system). Accordingly, Eqs. (25) and (28) cannot be treated in a simpler manner. An iterative scheme has to be applied. For the finite strain case, the application of the Newton–Raphson method, see Eq. (34), is similar to the small strain case. Here, the derivative of Eq. (16) is required, and the geometrical non-linear term resulting from the weak form has to be taken into consideration, see [18,23] for the necessary derivatives with respect to the displacements.

3.3 Diagonally implicit Runge–Kutta methods

The application of diagonally implicit Runge–Kutta methods has been successfully carried out in a couple of publications [12,18,20,22,28]. In this case, the Butcher-tableau of Fig. 1c contains diagonal elements. This yields

$$\boldsymbol{Q}_{ni} = \boldsymbol{q}_n + \Delta t_n \sum_{j=1}^{i} a_{ij} \boldsymbol{r}(T_{nj}, \boldsymbol{U}_{nj}, \boldsymbol{Q}_{nj}) = \boldsymbol{S}^{\mathrm{q}}_{ni} + \Delta t_n \, a_{ii} \boldsymbol{r}(T_{ni}, \boldsymbol{U}_{ni}, \boldsymbol{Q}_{ni}) \tag{37}$$

with the stage quantity

$$\boldsymbol{S}^{\mathrm{q}}_{ni} = \boldsymbol{q}_n + \Delta t_n \sum_{j=1}^{i-1} a_{ij} \dot{\boldsymbol{Q}}_{nj} \tag{38}$$

and the abbreviation of the stage derivative

$$\dot{\boldsymbol{Q}}_{ni} = \boldsymbol{r}(T_{ni}, \boldsymbol{U}_{ni}, \boldsymbol{Q}_{ni}). \tag{39}$$

In the ith stage, $\boldsymbol{S}^{\mathrm{q}}_{ni}$ is known. However, $\boldsymbol{Q}_{ni}$ in Eq. (37) depends on itself. In other words, at each stage T_{ni}, the system

$$\boldsymbol{L}(T_{ni}, \boldsymbol{U}_{ni}, \boldsymbol{Q}_{ni}) = \boldsymbol{Q}_{ni} - \boldsymbol{S}^{\mathrm{q}}_{ni} - \Delta t_n \, a_{ii} \boldsymbol{r}(T_{ni}, \boldsymbol{U}_{ni}, \boldsymbol{Q}_{ni}) = \boldsymbol{0} \tag{40}$$

together with the algebraic constraint

$$\boldsymbol{g}(T_{ni}, \boldsymbol{U}_{ni}, \boldsymbol{Q}_{ni}) = \boldsymbol{0} \tag{41}$$

has to be solved. This represents a system of coupled, non-linear equations for the unknown nodal displacements $\boldsymbol{U}_{ni}$ and the internal variables $\boldsymbol{Q}_{ni}$ at each stage T_{ni}. Again, a stiffly accurate method only requires the computations of s systems of non-linear Eqs. (40)–(41). The final values of the displacements and the internal variables are given by

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + \Delta t_n \sum_{i=1}^{s} b_i \dot{\boldsymbol{U}}_{ni} \tag{42}$$

$$\boldsymbol{q}_{n+1} = \boldsymbol{q}_n + \Delta t_n \sum_{i=1}^{s} b_i \dot{\boldsymbol{Q}}_{ni} \tag{43}$$

with

$$\dot{\boldsymbol{U}}_{ni} = \frac{\boldsymbol{U}_{ni} - \boldsymbol{S}^{\mathrm{u}}_{ni}}{\Delta t_n\, a_{ii}}, \quad \dot{\boldsymbol{Q}}_{ni} = \frac{\boldsymbol{Q}_{ni} - \boldsymbol{S}^{\mathrm{q}}_{ni}}{\Delta t_n\, a_{ii}}, \tag{44}$$

which are stored at each stage T_{ni}. Analogously to (38) one has to compute

$$\boldsymbol{S}^{\mathrm{u}}_{ni} = \boldsymbol{q}_n + \Delta t_n \sum_{j=1}^{i-1} a_{ij}\, \dot{\boldsymbol{Q}}_{nj} \tag{45}$$

In [19], it is shown that the coupled system (40)–(41) is mentioned to be solved by means of the Newton–Raphson method (NRM) that is in most situations not true. Instead, the Multilevel-Newton algorithm (MLNA) developed by Rabbat et al. [38], see [32] as well, is applied. In the case of the classical NR-method, no local iteration on Gauss-point level would occur. The MLNA is based on the implicit function theorem. This requires in each iterations step the computation of the non-linear Eq. (40) for given displacements. Since the internal variables and their ODEs are only formally assembled in order to see the underlying DAE-system (18) and (23), the system (40) decomposes to the computation of the internal variables on Gauss-point level. Frequently, the structure of the scheme cannot be identified, because it is common to follow the idea to reduce the number of unknowns on Gauss-point level by analytical considerations, see, for example, [18,24,27,36,43]. For instance, the integration step of the small strain viscoelasticity model (5),(9) and (10) is reducible to merely one equation with one unknown by a substitution technique, see [19]. Or the integration step of the complex finite strain model (11)–(15) can be manipulated to get three equations with three unknowns, see [18]. This essentially reduces the computational effort. Usually, the MLNA approach is faster than the NRM applied to the system (40)–(41), see investigations in [19]. For further efficiency considerations to solve the system (40)–(41) by iterative schemes, see [30].

The most simplest DIRK method is represented by the Backward-Euler method ($s = 1, c_1 = 1, a_{11} = 1, b_1 = 1$) that is only of order one. The combination of the Backward-Euler method (BE) and the MLNA yields exactly the same implementation of current implicit finite element programs if the constitutive model is integrated on Gauss-point level by a BE-integration step. Since the BE-method is embedded in the DIRK methods and in each stage T_{ni} the system of the non-linear equations looks equivalent (merely, $\boldsymbol{q}_n \to \boldsymbol{S}^{\mathrm{q}}_{ni}$ and $\Delta t_n \to a_{ii}\Delta t_n$ change), current non-linear finite element programs are easily changeable to high-order time integration, which is demonstrated in [5].

3.4 Rosenbrock-type methods

Rosenbrock-type methods (ROW methods) are one-step methods as well. They can be developed on the basis of DIRK methods, where only one Newton-iteration step is carried out. Additional ideas to retain stability properties and to reduce the effort have to be introduced, see both [14] for the basic ideas and its mathematical foundation as well as [23,26] for the application to finite elements. This leads to only one system of linear equation in each stage T_{ni} having the dimension n_{u} and not to a system of non-linear equations such as in Sects. 3.1–3.3. Before this resulting system is recapped, one issue has to be considered: HERK and DIRK methods consist of the by-product that the reaction forces at those nodes where displacements are prescribed are simple computable, see [29]. In the context of ROW methods, additional effort has to be done. This is performed using the method of Lagrange-multipliers. Nevertheless, all these aspects lead in each stage to a system of linear equations with the dimension n_{u} because a particular Schur-complement technique is applied. In other words, small systems of linear equations with several right-hand sides, $n_{\mathrm{q}} + 1$, on Gauss-point level have to be computed. In Table 1 the time-integration method of Rosenbrock-type without reaction force computation is summarized. Because of the complexity in the reaction force computation, it is referred to [23]. In Table 1, the abbreviations

$$\boldsymbol{b}^{q}_{ni} := \gamma_i \Delta t_n \frac{\partial \boldsymbol{r}}{\partial t} + \boldsymbol{r}\left(T_{ni}, \boldsymbol{S}^{\mathrm{q}}_{ni}, \boldsymbol{S}^{\mathrm{u}}_{ni}\right) + \sum_{j=1}^{i-1} \left(\frac{c_{ij}}{\Delta t_n}\right) \boldsymbol{V}^{q}_{nj}, \tag{46}$$

$$\boldsymbol{b}^{u}_{ni} := \gamma_i \Delta t_n \frac{\partial \boldsymbol{g}}{\partial t} + \boldsymbol{g}\left(T_{ni}, \boldsymbol{S}^{\mathrm{q}}_{ni}, \boldsymbol{S}^{\mathrm{u}}_{ni}\right), \tag{47}$$

Table 1 Rosenbrock-type methods in the framework of non-linear finite element analysis

Given:	Coefficients of the Rosenbrock-type method Initial- and boundary conditions, $\boldsymbol{y}(t_0) = \{\boldsymbol{Q}_0, \boldsymbol{u}_0\} = \boldsymbol{y}_0$	
Loop	over time-steps $n = 0, \ldots, N$ argument vector: $\boldsymbol{z}_n := (\boldsymbol{q}_n, \boldsymbol{u}_n)$	
	Loop over stages $i = 1, \ldots, s$	
	Compute on global level $T_{ni} = t_n + c_i \Delta t_n\,, \quad \boldsymbol{S}_{ni} = \boldsymbol{y}_n + \sum_{j=1}^{i-1} a_{ij} \boldsymbol{V}_{nj},$	
	Solve on local level $\left[\left.\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{q}}\right\|_{\boldsymbol{z}_n} - \hat{\gamma} \boldsymbol{I} \right] [\boldsymbol{X}_n^{\boldsymbol{M}} \| \boldsymbol{x}_{ni}^{v}] = \left[\left.\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{u}}\right\|_{\boldsymbol{z}_n} \| \boldsymbol{b}_{ni}^{q} \right]$	$\rightsquigarrow \boldsymbol{X}_n^{\boldsymbol{M}}, \boldsymbol{x}_{ni}^{v}$
	Solve on global level $\boldsymbol{K} \boldsymbol{V}_{ni}^{u} = -\boldsymbol{b}_{ni}^{u} + \frac{\partial \boldsymbol{g}(t_n, \boldsymbol{u}_n, \boldsymbol{q}_n)}{\partial \boldsymbol{q}} \boldsymbol{x}_{ni}^{v}$	$\rightsquigarrow \boldsymbol{V}_{ni}^{u}$
	Compute on global level $\boldsymbol{V}_{ni}^{q} = -\boldsymbol{x}_{ni}^{v} - \boldsymbol{X}_n^{\boldsymbol{M}} \boldsymbol{V}_{ni}^{u_a}$	$\rightsquigarrow \boldsymbol{V}_{ni}^{q}$
	Update $\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + \sum_{i=1}^{s} m_i \boldsymbol{V}_{ni}$	
	Estimate local integration error Compute new step-size If step is not accepted, repeat time-step $t_n \rightarrow t_{n+1}$ with reduced step-size Δt_n	

are introduced. $\gamma_i = \sum_{j=1}^{i} \gamma_{ij}$, $\hat{\gamma} = 1/(\gamma \Delta t_n)$ with the diagonal element γ of the γ_{ij}-matrix, m_i, and c_i are coefficients of the method. $\boldsymbol{K} \in \mathbb{R}^{n_u \times n_u}$ represents the stiffness matrix evaluated at time t_n. On element, that is, Gauss-point level, the matrix $\boldsymbol{X}_n^{\boldsymbol{M}}$ and the vector $\boldsymbol{x}_{ni}^{v}$ have to be computed by a system of n_q linear equations with 7 right-hand sides. This is symbolized by the notation $[\boldsymbol{X}_n^{\boldsymbol{M}} | \boldsymbol{x}_{ni}^{v}]$ indicating an assemblage of a matrix and a vector into a matrix. These quantities are assembled into a large matrix and a vector that are used for a matrix-vector product to obtain the stage vector $\boldsymbol{V}_{ni}^{q}$ on global level.

4 Numerical examples

The three one-step methods, HERK, DIRK, and ROW methods, will be compared at 3D inhomogeneous deformations. Special emphasis lies in the efficiency and accuracy of the entire procedure. Use is made of GiD pre- and post-processor as well as the inhouse finite element program TASA-FEM. The computations are carried out on a double quad-core computer with an Intel processor Q9650 (12M Cache, 3 GHz, 1,333 MHz FSB). Furthermore, the linear solver PARDISO, [40], addressing four cores is applied.

In view of comparing the procedures, we divide into both order considerations and step-size behavior as well as small and finite strain viscoelasticity. The chosen integration schemes are compiled in Table 2. The weighting factors of the methods are assembled in the Butcher-tableaus of Tables 7, and 8 in Appendix B. In this article, we restrict ourselves to methods of order one, two, and three.

Some remarks in view of efficiency improvements have to stated. The DIRK methods are optimized using a linear interpolation estimator for the starting vector of the Multilevel-Newton algorithm, see [30], and problem-adapted stress-algorithms leading to one non-linear equation for one unknown in the small strain case, see [19], and three non-linear equations in the finite strain example, [18]. In the ROW and HERK-methods, only the minimum of LU-decompositions are performed. All further computations of the linear systems requiring only back-substitutions as long as the coefficient matrices are constant. Moreover, the stress computations and the evaluations of the internal variables are done on Gauss-point level.

For the small strain viscoelasticity, the chosen material parameters are assembled in Table 3, whereas the parameters of the finite strain viscoelasticity model are compiled in Table 4. The problems under consideration are a distance spacer for small strain applications and a strut bearing for a finite strain application.

Table 2 Applied one-step methods with references in the context of ODE/DAE solution and applications within finite elements (PDAEs)

abbreviation	order p	stages s	reference
HERK1	1	1	Half-explicit Euler (HEE), Hairer and Wanner [14]
HERK2	2	2	Sofroniou and Spaletta [46]
HERK3	3	3	Bogacki and Shampine [6]
DIRK1	1	1	Backward-Euler (BE), Hairer and Wanner [14]; Hartmann [18]
DIRK2	2	2	Ellsiepen [11]; Ellsiepen and Hartmann [12]; Hartmann [18]
DIRK3	3	3	Alexander [1]; Cash [7]; Hartmann [18]
ROW1	1	1	Linear-implicit Euler (LIE), Hairer and Wanner [14]; Hartmann and Hamkar [23]
ROW2	2	3	ROS2S, Hamkar et al [17]
ROW3	3	4	ROSI2PW, Rang and Angermann [39]; Hartmann and Hamkar [23]

Table 3 Material parameters of small strain viscoelasticity model

K	G	$\hat{G}$	η_0	s_0
MPa	MPa	MPa	MPa s	MPa^{-1}
1,667	178.8	1500	1.3×10^5	1.1

Table 4 Material parameters of finite strain viscoelasticity model

K	c_{10}	c_{01}	α	μ	η_0	s
MPa	MPa	MPa	MPa	MPa	MPa s	MPa^{-1}
1,000	0.1788	0.1958	3.67×10^{-2}	0.2	180	10^{-3}

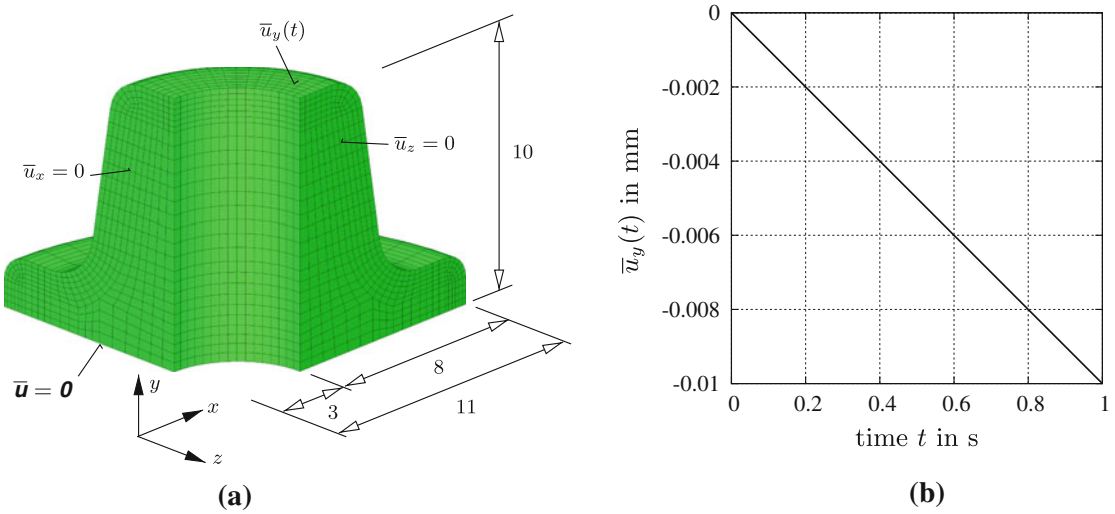

Fig. 2 Mesh and geometry of a quarter of a distance spacer pressed by a displacement $\overline{u}_y(t)$ on the top (measures in mm). **a** Boundary conditions. **b** Linear loading path

4.1 Order investigation of time-integration methods

4.1.1 Small strain viscoelasticity

First of all, the HERK-methods are compared to ROW and DIRK methods in view of their expected and achieved order. To this end, a distance spacer is meshed using eight-noded hexahedral elements with linear shape functions, see Fig. 2a. The distance spacer is glued at the bottom and loaded at the upper surface in y-direction, that is, it is linearly pressed at the top of the spacer, see Fig. 2b. Additionally, symmetry conditions are introduced at the cutting surfaces. The size of the problem is given by the number of unknown nodal displacements, $n_{\text{u}} = 22{,}001$, and the number of internal variables at all Gauss-points $n_{\text{Q}} = 330{,}240$.

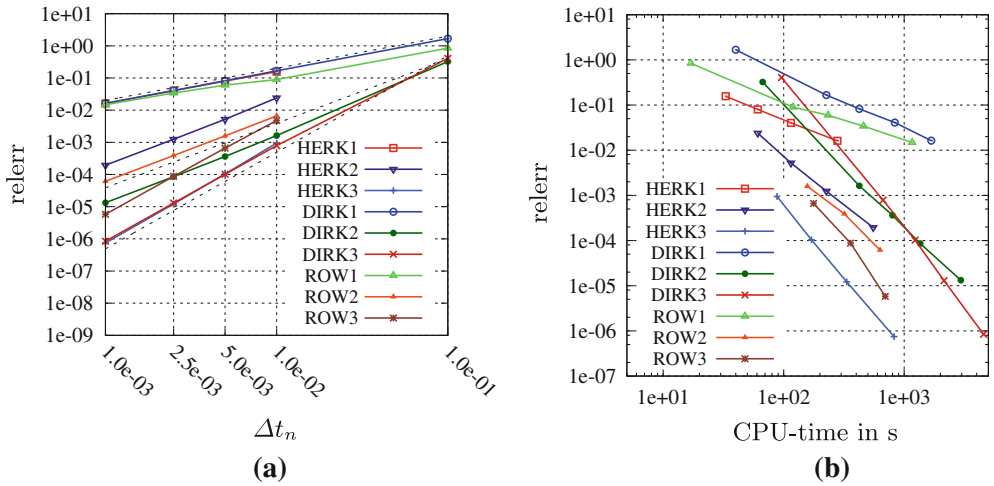

Fig. 3 Constant step-size behavior of the methods in Table 2 applied to problem 2(a). **a** Convergence order. **b** Efficiency

For the order considerations, a relative error of the internal variables is compared to the order one, two, and three methods of Table 2. The reference solution is given by a fifth order method HEDOPRI (half-explicit Dormand and Prince method, see [9]), with a constant step-size of $\Delta t = 10^{-3}$ s. The relative error is defined by the maximum relative error over all points in time t_n:

$$\text{relerr} := \max_n \left(\frac{\|\boldsymbol{q}_n^{\text{ref}} - \boldsymbol{q}_n\|}{\|\boldsymbol{q}_n^{\text{ref}}\|} \right). \tag{48}$$

HERK1 defines the half-explicit Euler method ($p = 1$ $s = 1$); HERK2 and HERK3 are stiffly accurate second- and third-order methods. Figure 3a shows that for a sufficient small time step, the expected order of all methods is achieved, that is, there have no order reduction phenomena been observed, such as in plasticity problems with yield function, see, for example, [12,21,28]. For larger step-sizes, only the half-explicit Euler (HERK1) and linear-implicit Euler (ROW1) as well as the DIRK method lead to results. In the other cases, the local errors are drastically increased leading to inappropriate results or failing coordinate transformations. Additionally, it is observable that the chosen ROW3 method shows errors within the range of the second-order methods (although the order is obtained). These observations might be caused by the limited stability properties of some integrators.

Remark 1 The stability criteria of the methods developed for the ODE-case are, here, transferable to the DAE-case caused by the property (29). In order to show this, Eq. (29) can be solved for a regular tangential stiffness matrix $\boldsymbol{K}$ to obtain

$$\hat{\boldsymbol{u}}(t, \boldsymbol{q}) = \boldsymbol{K}^{-1}\{\overline{\boldsymbol{p}}(t) + \boldsymbol{K}_{\text{q}}\boldsymbol{q} - \overline{\boldsymbol{K}}\,\overline{\boldsymbol{u}}(t)\}.$$

This expression is inserted into the evolution Eqs. (23) so that we arrive at

$$\dot{\boldsymbol{q}}(t) = \boldsymbol{r}(t, \hat{\boldsymbol{u}}(t, \boldsymbol{q}), \boldsymbol{q}(t)).$$

If $\boldsymbol{r}$ is linear in $\boldsymbol{u}$ and $\boldsymbol{q}$, which is the case of linear viscoelasticity, a linear ODE results. However, for the general non-linear case (18), a stability criteria is not available. Since it is not the goal of this manuscript to develop such a criterion, only numerical results are interpreted. It must be noted that step-size control techniques generally do not overcome the inherent disadvantage of explicit methods. However, for appropriate error tolerances, the errors are controlled.

If we look at the computational costs for constant step-sizes in Fig. 3b of the methods compiled in Table 2, the following results are observable. The HERK-methods are much faster than the corresponding DIRK method of the same order. However, for nearly the same low accuracy and the same constant step-size, the linear-implicit Euler method (ROW1) saves more than 70% computational time. Again it must be emphasized that we do not solve the linear systems (33) directly but apply a non-linear solver to it according to Eq. (34) requiring two computations of the linear system, that is, it is not optimally implemented (one can guess that additional 30–40% computational costs can be saved). For a relative error of 10^{-3}, HERK3 is the fastest followed by ROW3 and ROW2. Classical implicit formulations are much more expensive to reach the same accuracy.

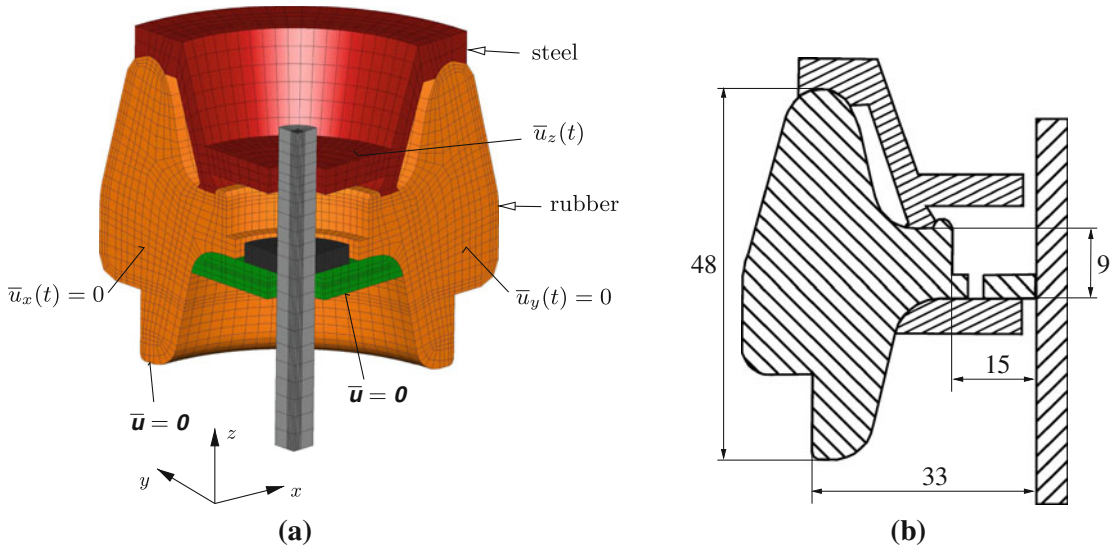

Fig. 4 Load and boundary conditions of a strut bearing made of carbon black-filled elastomer. **a** Mesh and boundary conditions of a strut bearing. **b** Cross section with rough geometrical dimensions (measures in mm)

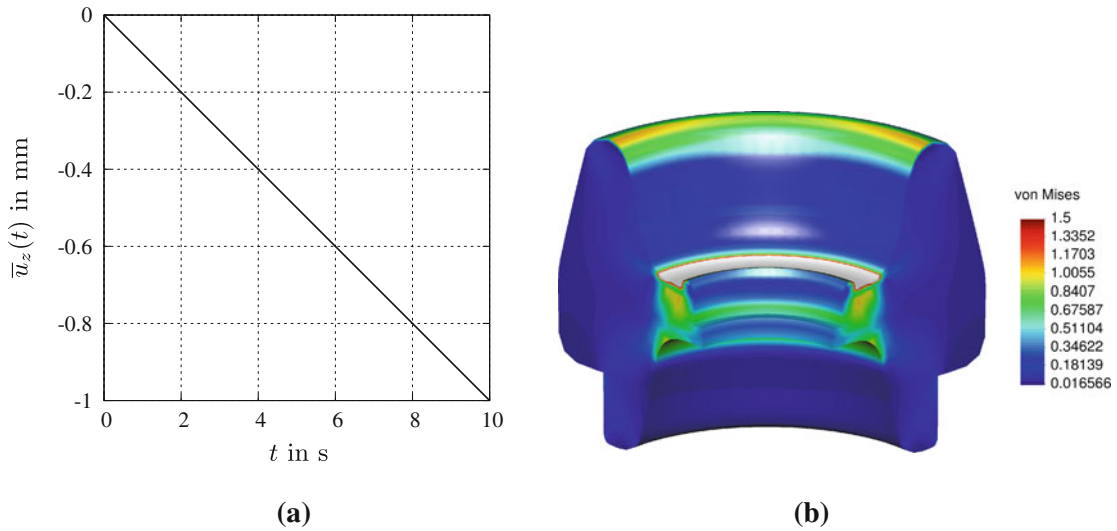

Fig. 5 Loading path and resulting stresses in the strut bearing example. **a** Loading path of strut bearing example. **b** Von Mises stress distribution caused by fixation of the strut bearing (stresses in MPa)

4.1.2 Finite strain viscoelasticity

For the case of a much more non-linear application, a strut bearing is investigated. The discretization is shown in Fig. 4a. The load $\overline{u}_z(t)$ describes the application of a metal nut onto a steel plate ($E = 210{,}000\,\text{MPa}$, $\nu = 0.3$) representing the fixation at the chassis. Here, we have $n_\text{u} = 49{,}079$ unknown nodal displacements and $n_\text{Q} = 708{,}480$ internal variables in $n_\text{e} = 14{,}760$ elements. Furthermore, $n_\text{p} = 3{,}496$ is given. The maximum strains, here the norm of the eigenvalues of logarithmic strain tensor $\ln \mathbf{U}$, $\mathbf{U}$ defines the right stretch tensor of the polar decomposition of the deformation gradient, is chosen, lie approximately at $\max(\ln \lambda_k) \approx 16\%$. The results described by von Mises stresses are shown in Fig. 5b indicating the local behavior of the loading to a large extent.

Again, the order of the methods is achieved, see Fig. 6a. Only for small step-sizes of the HERK3 and DIRK3 methods, the order seems not to be reached. However, in this range, the accuracy is close to the machine precision so that these points have to be carefully treated. In this example, the HERK-methods require in each time step the solution of a system of non-linear equations of dimension n_u. The computational expenditure

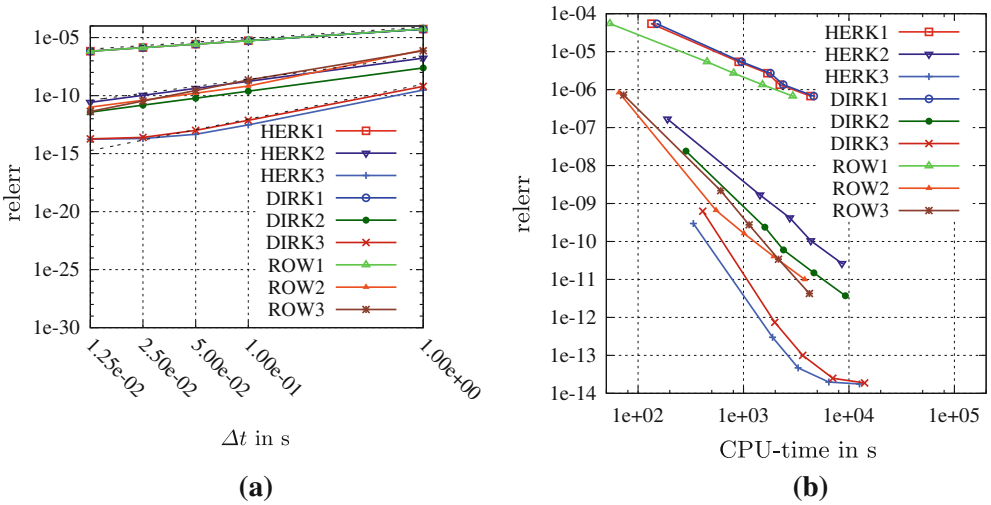

Fig. 6 Constant step-size behavior (**a** order consideration and **b** efficiency) of the finite strain viscoelasticity computations of a strut bearing

on Gauss-point level is larger than in the small strain case caused by the geometrical non-linearity. There are global iterations of the HERK-method in the strut bearing example necessary, and in each iterative solution, one LU-decomposition is required. The DIRK/MLNA approach requires the solution of three equations with three unknowns at each Gauss-point, see [18]. In view of the efficiency, see Fig. 6b, one can conclude that ROW methods are much faster than the HERK/NRM and DIRK/MLNA schemes, even if 3 stages for an order 2 method are required.

4.2 Step-size behavior and efficiency

In the next investigation, the step-size controlled behavior is studied to see the real behavior and the overall efficiency of the methods. Since there is no embedded method for the order one schemes, only order two and three procedures are applied. In the forthcoming examples, long-time loading paths are considered because non-stepsize-controlled procedures are unsuited for increasing the step-size in an appropriate manner which fail caused by the extremely long computing times.

4.2.1 Small strain viscoelasticity

In the first example, the step-size behavior of the time-adaptive procedures using local error estimations and embedded one-step methods is investigated, see for the origin [16] and for finite element applications [8,12,18,23]. For all methods, the same absolute and relative error tolerances for the displacements ($\varepsilon^u_{\text{abs}} = 10^{-4}$, $\varepsilon^u_{\text{rel}} = 10^{-5}$) and the internal variables ($\varepsilon^q_{\text{abs}} = 10^{-5}$, $\varepsilon^q_{\text{rel}} = 10^{-6}$) are used in order to estimate the new step-size. In respect of the definition of the error tolerances, see [12]. In the case of the proposed HERK-methods, only the errors of the differential part (internal variables) are used for step-size control.

The example is one-half of the distance spacer, which is linearly pressed and subsequently sheared. Both processes are followed by a long-time holding time, where classical finite elements either fail or the computational times are very high caused by the missing step-size control technique. The prescribed displacement boundary conditions in Fig. 7 are given by

$$\overline{u}_y(t) = \begin{cases} v_y t & \text{for } 0 \le t \le 10 \text{ s} \quad \text{with } v_y = -10^{-3}\text{mm/s} \\ \hat{u}_y \text{ mm} & \text{for } 10 \text{ s} \le t \le 1{,}000 \text{ s} \quad \text{with } \hat{u}_y = -10^{-2}\text{mm} \end{cases}$$

and

$$\overline{u}_x(t) = \begin{cases} 0 & \text{for } 0 \le t \le 10 \text{ s} \\ v_x(t - t_0) & \text{for } 10 \text{ s} \le t \le 20 \text{ s} \quad \text{with } v_x = 5 \times 10^{-3} \text{ mm/s and } t_0 = 10 \text{ s} \\ \hat{u}_x & \text{for } 20 \text{ s} \le t \le 1{,}000 \text{ s} \quad \text{with } \hat{u}_x = 5 \times 10^{-2} \text{ mm} \end{cases}$$

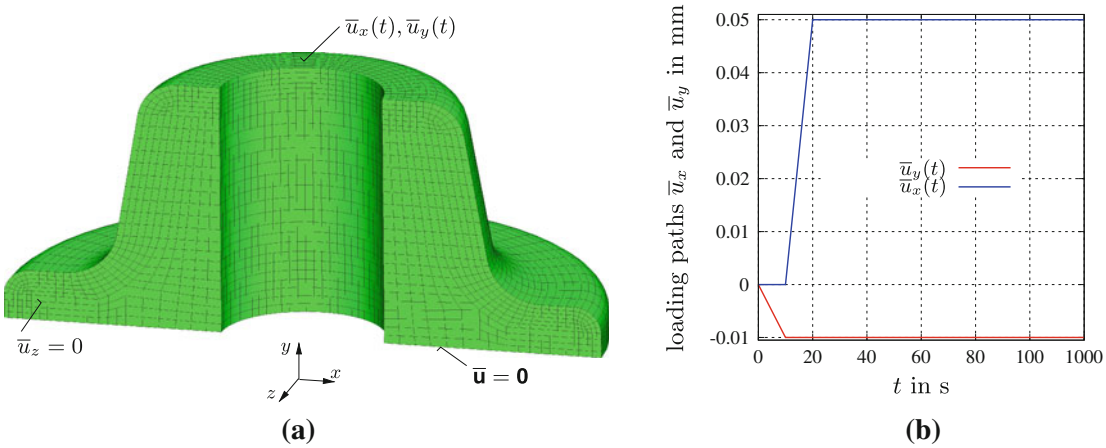

Fig. 7 Half of a distance spacer under compression and shear loading. **a** Boundary conditions ($n_{\mathrm{u}} = 43{,}553$, $n_{\mathrm{p}} = 3{,}868$, $n_{\mathrm{Q}} = 660{,}480$). For the geometrical data see Fig. 2. **b** Compression and shear loading path at the top of the spacer

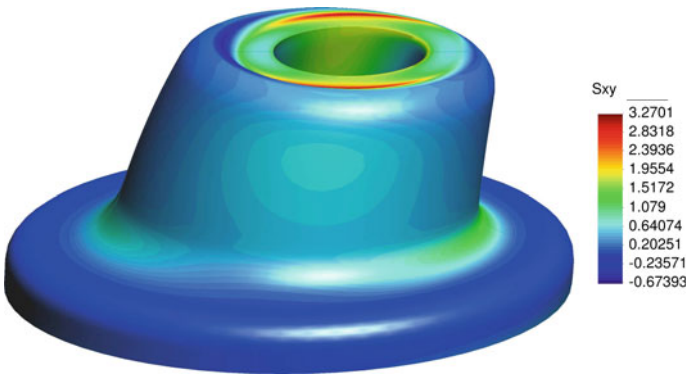

Fig. 8 Shear stress distribution (deformation magnification factor 50). Stresses are given in MPa

Figure 8 shows exemplarily the shear stress distribution and the resulting deformation magnified by the factor 50 at $t = 1{,}000$ s. The step-size behavior is given in Fig. 9, where in Fig. 9a the entire process and in Fig. 9b only the first 40 s are shown. The rapid decrease at $t = 10$ s and $t = 20$ s is caused by the kink in the loading curves. There the program starts with the user-estimated starting step-size $\Delta t_0 = 10^{-3}$ s. Obviously, the step-sizes are smallest for the HERK2-method. However, if one looks at the CPU-times the HERK-schemes, they lead to the fastest computations, see Table 5. HERK and ROW methods require only 14–22% of the computational time of the 3rd order DIRK method. Although, the HERK2-approach requires smaller step-sizes than all other methods, it is more efficient than most other methods caused by the fact that only one LU-decomposition is necessary for the entire integration process. In other words, the computational times can essentially be reduced by choosing an appropriate time-integration scheme.

4.2.2 Finite strain viscoelasticity

In the finite strain viscoelasticity example of the strut bearing, the loading path of linear pressing within 10 s and subsequent holding time of 90 s is considered, see Fig. 10a. For all methods, the same absolute and relative error tolerances for the internal variables ($\varepsilon^{q}_{\mathrm{abs}} = 10^{-5}$, $\varepsilon^{q}_{\mathrm{rel}} = 10^{-6}$) are used in order to estimate the new step-size. The DIRK and ROW methods additionally use the error of displacements to predict the new step-size ($\varepsilon^{u}_{\mathrm{abs}} = 10^{-4}$, $\varepsilon^{u}_{\mathrm{rel}} = 10^{-5}$). In this case, the advantage of HERK-methods to evaluate only one time the LU-decomposition is not possible. In order to satisfy the error tolerance, the half-explicit methods require more computational time than the implicit ones, see Fig. 10b. However, if one looks at ROW methods, their

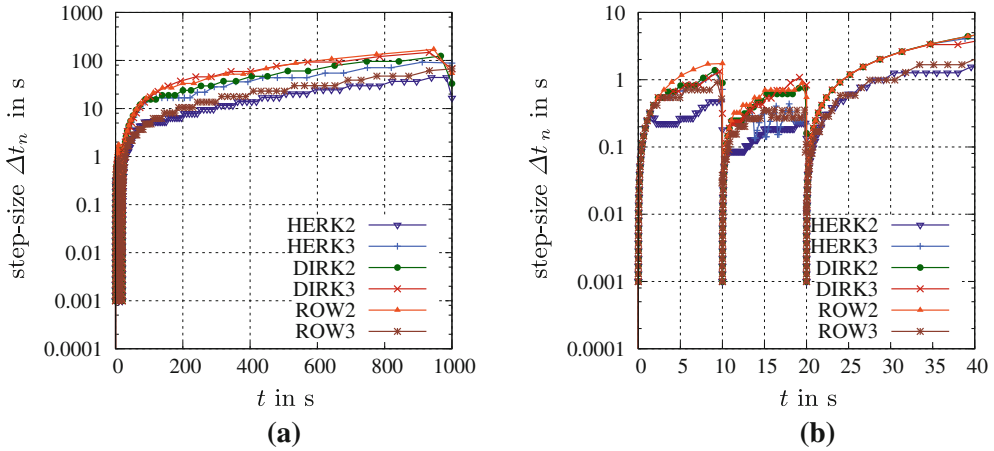

Fig. 9 Time step-size behavior and computational costs. **a** Step-size behavior of the loading process of Fig. 7b. **b** Zoom of the step-size behavior

Table 5 Computational costs of the three one-step schemes (small strain example)

Method	No. of time steps	CPU time	Eff. factor
HERK2	277	373 s	0.14
HERK3	172	348 s	0.13
DIRK2	140	1,869 s	0.68
DIRK3	135	2,754 s	1
ROW2	121	518 s	0.19
ROW3	149	596 s	0.22

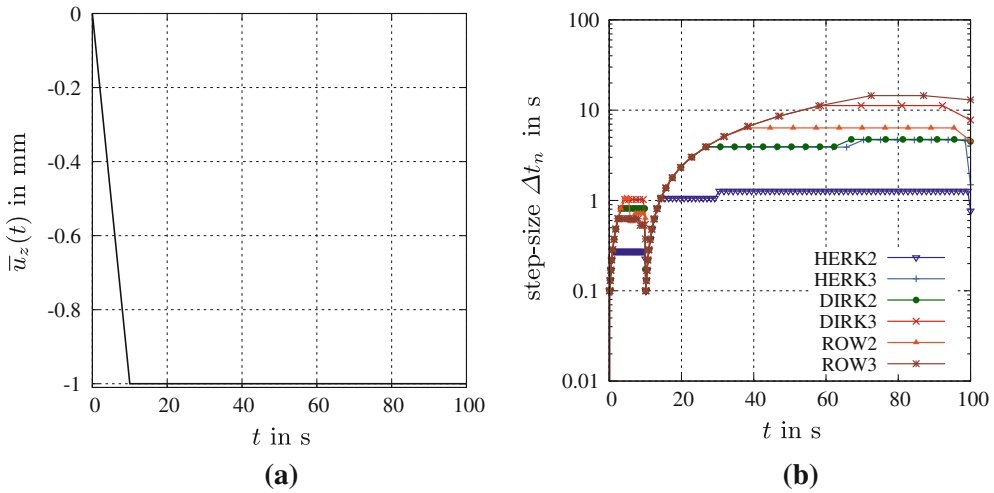

Fig. 10 **a** Loading path and **b** step-size behavior of the large strain example (strut bearing in Fig. 4)

efficiency is superior to the other methods, see Table 6. This is caused by one LU-decomposition at each time t_n and additional back-substitutions at the stage times T_{ni}, the missing iterations, globally and locally.

Table 6 Computational costs of the three one-step schemes (finite strain example)

Method	No. of time steps	CPU time	Eff. factor
HERK2	122	1,456 s	1.55
HERK3	55	1,014 s	1.08
DIRK2	54	787 s	0.84
DIRK3	43	938 s	1
ROW2	48	230 s	0.25
ROW3	44	243 s	0.26

5 Conclusions

Three different one-step methods, half-explicit Runge–Kutta methods (HERK), diagonally implicit Runge–Kutta methods (DIRK), and Rosenbrock-type methods (ROW) are compared for finite element computations of problems in quasi-static solid mechanics, where the constitutive models are of rate-type. In the general case, HERK-methods lead to a system of non-linear equations with n_{u} unknown nodal displacements in each point in time. The internal variables on Gauss-point level are computed by an explicit time-integration step. The advantage lies in the fact that a very easy implementation of the constitutive model is possible, which can be seen in the context of "rapid prototyping" of models. The drawback is the stability property, that is, only very small step-sizes are allowed. In the case of viscoelasticity for small and finite deformation in combination of a step-size control technique, a sufficient applicability is observed. However, for a save implementation of viscoplasticity models with yield function (case distinction), it seems to be not feasible in our first investigations (which are not summarized here). The main advantage, however, can be seen in the efficiency for models, which are linear in the displacements, because only one LU-decomposition has to be carried out for the entire integration process. All further computations of linear systems with the same coefficient matrix require merely back-substitutions. Thus, in this case, the overall computation time is better than for fully implicit time-integration schemes. In the cases where the problem in non-linear in the displacement field, HERK-methods combined with the Newton–Raphson method (NR-method) are, in view of efficiency, equivalent to DIRK methods combined with a Multilevel-Newton algorithm (MLNA) based on a "problem-adapted stress algorithm", that is, additional analytical reduction of unknowns on Gauss-point level. That means that a fast implementation is possible, but for a more stable implementation, the DIRK/MLNA approach should be preferred.

DIRK/MLNA have their advantage in their stability (stable in the sense of mathematical stability criteria and in the sense of computations leading to a result close to the exact solution). A further advantage lies in the fact that even if some programming errors are incorporated into the functional matrices, only the convergence of the MLNA is affected and not the result itself. The disadvantage lies in the fact of computational costs and implementation aspects (costly derivatives – consistent tangent operator). Here, methods for reducing the effort of the resulting coupled non-linear equations have to be looked for (estimators of starting vector of the MLNA, optimal solver of the coupled non-linear system and of the resulting systems of linear equations, problem-adapted time integration).

Rosenbrock-type methods are very efficient schemes because they lead only to linear systems of equations, which require only back-substitutions at stages greater than one. Therefore, they are really fast in comparison to DIRK methods. The stability properties are appropriate for the underlying PDAE-systems. The behavior in yield function based models is unknown up to now. The disadvantage can be seen for cases where programming errors influence the functional matrices yielding in most cases a large difference to the exact solution. Here, the code verification aspect is not negligible.

Acknowledgments We would like to thank the German Research Foundation (DFG, grant no. HA2024/7-1) and the "Simulationswissenschaftliches Zentrum" of the Clausthal University of Technology for financial support.

Table 7 Butcher-tableaus of second-order HERK and DIRK method

0	0	0	0
1	1	0	0
1	$\frac{1}{2}$	$\frac{1}{2}$	0
	$\frac{1}{2}$	$\frac{1}{2}$	0
	1	$-\frac{1}{6}$	$\frac{1}{6}$

(a) HERK2 [46]

α	α	0	
1	$1-\alpha$	α	
	$1-\alpha$	α	
	$1-\hat{\alpha}$	$\hat{\alpha}$	

(b) DIRK2 [12], $\alpha = 1 - \frac{1}{2}\sqrt{2}$, $\hat{\alpha} = 2 - \frac{5}{4}\sqrt{2}$

Table 8 Butcher-tableaus of third-order HERK and DIRK method

0	0	0	0	0
$\frac{1}{2}$	$\frac{1}{2}$	0	0	0
$\frac{3}{4}$	0	$\frac{3}{4}$	0	0
1	$\frac{2}{9}$	$\frac{1}{3}$	$\frac{4}{9}$	0
	$\frac{2}{9}$	$\frac{1}{3}$	$\frac{4}{9}$	0
	$\frac{7}{24}$	$\frac{1}{4}$	$\frac{1}{3}$	$\frac{1}{8}$

(a) HERK3 [6]

γ	γ	0	0	
δ	$\tau-\gamma$	γ	0	
1	α	β	γ	
	α	β	γ	
	$\hat{\alpha}$	$\hat{\beta}$	0	

$\gamma = 0.4358665215084580$
$\tau - \gamma = 0.2820667392457705$
$\alpha = 1.2084966491760101$
$\beta = -0.6443631706844691$
$\delta = 0.7179332607542295$
$\hat{\alpha} = 0.7726301276675511$
$\hat{\beta} = 0.2273698723324489$

(b) DIRK3: Cash's method [7]

Appendix

A Butcher arrays

In this appendix the Butcher arrays, that is, the coefficients of the underlying second- and third-order methods are compiled in Tables 7 and 8. In view of the HERK-methods it should be noted that the coefficients were developed for the ODE-case. In this article, these methods are extended to the DAE-context mentioned above.

References

1. Alexander, R.: Diagonally implicit Runge–Kutta methods for stiff O.D.E.'s. SIAM J. Numer. Analy. **14**, 1006–1021 (1977)
2. Arnold, M.: Half-explicit Runge–Kutta methods with explicit stages for differential-algebraic systems of index 2. BIT **38**(3), 415–438 (1998)
3. Arnold, M., Strehmel, K., Weiner, R.: Half-explicit Runge–Kutta methods for semi-explicit differential-algebraic equations of index 1. Numer. Math. **64**, 409–431 (1993)
4. Beyn, W., Thummler, V.: Freezing solutions of equivariant evolution equations. SIAM J. Appl. Dyn. Syst. **3**(2), 85–116 (2004)
5. Birken, P., Quint. K.J., Hartmann, S., Meister, A.: A time-adaptive fluid-structure interaction method for thermal coupling. Comput. Vis. Sci. (2011). doi:10.1007/s00791-010-0150-4:1-10
6. Bogacki, P, Shampine, L.F.: A 3(2) pair of Runge–Kutta formulas. Appl. Math. Lett. **2**, 321–325 (1989)
7. Cash, J.R.: Diagonally implicit Runge–Kutta formulae with error estimates. J. Inst. Math. Appl. **24**, 293–301 (1979)
8. Diebels, S., Ellsiepen, P., Ehlers, W.: Error-controlled Runge–Kutta time integration of a viscoplastic hybrid two-phases model. Tech. Mech. **19**, 19–27 (1999)
9. Dormand, J.R., Prince, P.J.: A family of embedded Runge–Kutta formulae. J. Comput. Appl. Math. **6**, 19–26 (1980)
10. Eckert, S., Baaser, H., Gross, D., Scherf, O.: A BDF2 integration method with stepsize control for elastoplasticity. Comput. Mech. **34**(5), 377–386 (2004)
11. Ellsiepen, P.: Zeit- und ortsadaptive Verfahren angewandt auf Mehrphasenprobleme poröser Medien. Doctoral thesis, Institute of Mechanics II, University of Stuttgart, report No. II-3 (1999)
12. Ellsiepen, P., Hartmann, S.: Remarks on the interpretation of current non-linear finite-element-analyses as differential-algebraic equations. Int. J. Numer. Methods Eng. **51**, 679–707 (2001)
13. Fritzen, P.: Numerische Behandlung nichtlinearer Probleme der Elastizitäts- und Plastizitätstheorie. Doctoral thesis, Department of Mathematics, University of Darmstadt (1997)
14. Hairer, E., Wanner, G.: Solving Ordinary Differential Equations II. 2nd edn. Springer, Berlin (1996)
15. Hairer, E., Lubich, C., Roche, M.: The Numerical Solution of Differential-Algebraic Systems by Runge-Kutta Methods. Springer, Berlin (1989)
16. Hairer, E., Norsett, S.P., Wanner, G.: Solving Ordinary Differential Equations I, 2nd edn. Springer, Berlin (1993)
17. Hamkar, A.W., Hartmann, S., Rang, J.: A stiffly accurate Rosenbrock-type method of order 2. Appl. Numer. Math. (2010) (in review)

18. Hartmann, S.: Computation in finite strain viscoelasticity: finite elements based on the interpretation as differential-algebraic equations. Comput. Methods Appl. Mech. Eng. **191**(13–14), 1439–1470 (2002)
19. Hartmann, S.: A remark on the application of the Newton–Raphson method in non-linear finite element analysis. Comput. Mech. **36**(2), 100–116 (2005)
20. Hartmann, S.: A thermomechanically consistent constitutive model for polyoxymethylene: experiments, material modeling and computation. Arch. Appl. Mech. **76**, 349–366 (2006)
21. Hartmann, S., Bier, W.: High-order time integration applied to metal powder. Int. J. Plast. **24**(1), 17–54 (2008)
22. Hartmann, S., Bier, W.: High-order time integration applied to metal powder plasticity. Int. J. Plast. **24**(1), 17–54 (2008)
23. Hartmann, S., Hamkar, A.W.: Rosenbrock-type methods applied to finite element computations within finite strain viscoelasticity. Comput. Methods Appl. Mech. Eng. **199**(23–24), 1455–1470 (2010)
24. Hartmann, S., Haupt, P.: Stress computation and consistent tangent operator using non-linear kinematic hardening models. Int. J. Numer. Methods Eng. **36**, 3801–3814 (1993)
25. Hartmann, S., Neff, P.: Polyconvexity of generalized polynomial-type hyperelastic strain energy functions for near-incompressibility. Int. J. Solids Struct. **40**(11), 2767–2791 (2003)
26. Hartmann, S., Wensch, J.: Finite element analysis of viscoelastic structures using Rosenbrock-type methods. Comput. Mech. **40**, 383–398 (2007)
27. Hartmann, S., Lührs, G., Haupt, P.: An efficient stress algorithm with applications in viscoplasticity and plasticity. Int. J. Numer. Methods Eng. **40**, 991–1013 (1997)
28. Hartmann, S., Quint, K.J., Arnold, M.: On plastic incompressibility within time-adaptive finite elements combined with projection techniques. Comput. Methods Appl. Mech. Eng. **198**, 178–193 (2008)
29. Hartmann, S., Quint, K.J., Hamkar, A.W.: Displacement control in time-adaptive non-linear finite-element analysis. J. Appl. Math. Mech. **88**(5), 342–364 (2008)
30. Hartmann, S., Duintjer Tebbens, J., Quint, K.J., Meister, A.: Iterative solvers within sequences of large linear systems in non-linear structural mechanics. J. Appl. Math. Mech. (ZAMM) **89**(9), 711–728 (2009)
31. Hiley, R.A., Rouainia, M.: Explicit Runge–Kutta methods for the integration of rate-type constitutive equations. Comput. Mech. **42**(1), 53–66 (2008)
32. Hoyer, W., Schmidt, J.W.: Newton-type decomposition methods for equations arising in network analysis. ZAMM Zeitschrift für Angewandte Mathematik Und Mechanik **64**, 397–405 (1984)
33. Lion, A.: On the large deformation behaviour of reinforced rubber at different temperatures. J. Mech. Phys. Solids **45**, 1805–1834 (1997)
34. Liu, C.H., Hofstetter, G., Mang, H.A.: 3d finite element analysis of rubber-like materials at finite strains. Eng. Comput. **11**, 111–128 (1994)
35. Lubliner, J.: A model of rubber viscoelasticity. Mech. Res. Commun. **12**, 93–99 (1985)
36. Lührs, G., Hartmann, S., Haupt, P.: On the numerical treatment of finite deformations in elastoviscoplasticity. Comput. Methods Appl. Mech. Eng. **144**, 1–21 (1997)
37. Murua, A.: Partitioned half-explicit Runge–Kutta methods for differential-algebraic systems of index 2. Computing **59**(1), 43–61 (1997)
38. Rabbat, N.B.G., Sangiovanni-Vincentelli, A.L., Hsieh, H.Y.: A multilevel Newton algorithm with macromodeling and latency for the analysis of large-scale nonlinear circuits in the time domain. IEEE Trans. Circuits Syst. **26**, 733–740 (1979)
39. Rang, J., Angermann, L.: New Rosenbrock methods of order 3 for PDAEs of index 2. In: Proceedings of Equadiff, 11, pp 385–394. (2005)
40. Schenk, O., Gärtner, K.: Solving unsymmetric sparse systems of linear equations with Pardiso. Future Gener. Comput. Syst. **20**(3), 475–487 (2004). doi:10.1016/j.future.2003.07.011
41. Scherf, O.: Numerische Simulation inelastischer Körper. Fortschritt-Berichte VDI, Reihe 20 (Rechnerunterstützte Verfahren) Nr.321, VDI-Verlag, Düsseldorf (2000)
42. Schropp, J.: Geometric properties of Runge–Kutta discretizations for index 2 differential algebraic equations. SIAM J. Numer. Anal. **40**(3), 872–890 (2003)
43. Simo, J.C., Taylor, R.L.: Consistent tangent operators for rate-independent elastoplasticity. Comput. Methods Appl. Mech. Eng. **48**, 101–118 (1985)
44. Simo, J.C., Taylor, R.L.: Quasi-incompressible finite elasticity in principal stretches. Continuum basis and numerical algorithms. Comput. Methods Appl. Mech. Eng. **85**, 273–310 (1991)
45. Simo, J.C., Taylor, R.L., Pister, K.S.: Variational and projection methods for the volume constraint in finite deformation elasto-plasticity. Comput. Methods Appl. Mech. Eng. **51**, 177–208 (1985)
46. Sofroniou, M., Spaletta, G.: Construction of explicit Runge–Kutta pairs with stiffness detection. Math. Comput. Model. **40**, 1157–1169 (2004)
47. Strehmel, K., Weiner, R.: Numerik gewöhnlicher Differentialgleichungen. Teubner Verlag, Stuttgart (1995)
48. Wittekindt, J.: Die numerische Lösung von Anfangs-Randwertproblemen zur Beschreibung inelastischen Werkstoffverhaltens. Doctoral thesis, Department of Mathematics, University of Darmstadt (1991)