

Automated independent coordinates' switching for the solution of stiff DAEs with the linearly implicit Euler method

Mariano Carpinelli¹ · Marco Gubitosa¹ ·
Domenico Mundo² · Wim Desmet³

Received: 29 August 2014 / Accepted: 12 March 2015 / Published online: 29 April 2015
© Springer Science+Business Media Dordrecht 2015

Abstract The real-time (RT) integration of the equations of motion describing complex mechanical systems requires the employment of proper integration schemes which must be able to handle the time integration at each time step in less than an a priori fixed sampling time interval. The linearly implicit Euler method has been successfully employed for the RT integration of large stiff differential algebraic systems of equations (DAEs) which typically arise from the complex mechanical systems of interest in practical applications. The current industry demand is to further increase the degree of complexity of multibody models employed in RT applications, pushing researchers to improve the efficiency of the currently available integration methods. In this paper, we investigate the improvements in the efficiency of the linearly implicit Euler method coming from the conversion of the equations of motion at each time step from a dependent to an independent coordinates' formulation. The automatic switching from dependent to independent coordinates is achieved exploiting the properties of the matrix \mathbf{R} whose columns represent a basis of the null-space of the constraint Jacobian matrix. A non-iterative projection method is also applied in order to avoid the drift-off from the constraint conditions.

Keywords Multi-body dynamics · Real-time simulations · Independent coordinates' formulation · Vehicle dynamics

✉ M. Carpinelli
mariano.carpinelli@unical.it

M. Gubitosa
marco.gubitosa@siemens.com

D. Mundo
domenico.mundo@unical.it

W. Desmet
wim.desmet@kuleuven.be

¹ Siemens Industry Software NV, Interleuvenlaan 68, 3001 Leuven, Belgium

² Department of Mechanical Engineering, University of Calabria, 87036 Rende, Italy

³ Production Engineering, Machine Design and Automation (PMA) Section, KU Leuven, Celestijnenlaan 300, 3001 Leuven, Belgium

1 Introduction

The need for an efficient numerical integration of the equations of motion (EOMs) describing the complex multibody systems employed in vehicle dynamics applications has pushed the research on RT simulations in the last 20 years. Hardware-in-the-loop and software-in-the-loop (HIL, SIL) simulations as well as man-in-the-loop (MIL) applications, which have become standard tools for the design and the development of new vehicles, impose stringent requirements in terms of integration performances. This is dictated by the fact that real hardware must be interfaced to the mathematical models of the full vehicle or of one of its subsystems (e.g. suspension system and steering system). The electronic control units (ECUs) represent a typical example of the real hardware which is usually tested by means of HIL simulations in automotive applications. Since their control logics are generally not available for commercial reasons, a full virtual prototyping of the mechatronic system is not feasible, causing HIL simulations to be the only tool for analysing the correct interaction between the control system and the mechanical system. The frequency rate of the hardware embedded in the HIL simulation determines the frequency at which signals must be exchanged with the mathematical multibody model of the mechanical system, the current standard for automotive applications being 1 kHz. This determines the crucial requirement for HIL simulations, i.e. the integration process must provide the states of the mechatronic system at the current time step in less than 1 ms.

For a reliable real time simulation to take place, the number of operations carried out by the integration algorithm at each time step must be known a priori. Implicit schemes do not provide such a property since they compute the states of the system at the next time step by iteratively solving a non-linear system of equations. Explicit integration methods on the contrary rely on a fixed number of operations at each time step. Among them, low order schemes are generally preferred in RT applications due to the low number of operations executed at each time step. For this reason the explicit Euler method has been extensively used in HIL simulations. However, it can be only used for the time integration of non-stiff ODE systems, i.e. only the integration of the EOMs related to simplified vehicle models without stiff force elements and ideal joints connections can be addressed using this method. Numerical stiffness in the EOMs can be also caused by the modelling of subsystems with fast dynamics like electric circuits and controllers, which must be removed as well for the explicit Euler method to be effective.

The use of such simplified models in HIL simulations may not furnish accurate insights on the actual dynamic interaction between the controls and the mechanical systems. Moreover, a pressing demand in the automotive industry is pushing towards a reduction of the gap between complex, multi-domain, high-fidelity models developed during the design of mechanical systems and drastically simplified concept models employed in the design of control systems. Indeed, the simplification process required to transform a high-fidelity multibody model into one suitable for integration by means of the explicit Euler method is still a cumbersome and time consuming task. It involves the preparation of proper kinematics and compliance look-up tables (K&C tables) in order to take into account the characteristics of the suspension's linkages without retaining all the suspension links and the stiff bushing connections. Kinematic closed loop (i.e. the leverage in the steering system) must be replaced by kinematic look-up tables while open kinematic loops with ideal joints connections must be expressed in relative coordinates in order to avoid differential algebraic equations (DAEs). On top of that, every modification in the starting high-fidelity multibody model (e.g. a change in the positions of the suspension's hard points) requires the regeneration of the affected look-up tables in the simplified model.

To avoid all problems implied in the model simplification process, current research is focusing on the direct use of high-fidelity MBS models for HIL simulations [1–4]. In order to achieve RT integration of the EOMs associated to high-fidelity multibody systems containing stiff-force elements and ideal joints, the linearly implicit Euler method has been proposed which combines the efficiency of the explicit Euler method and the stability properties of the implicit Euler method [4, 5]. A great deal of research has been carried out to enhance the performances of the linearly implicit Euler method with the final aim of make the RT integration of large stiff DAE systems of equations arising from extremely complex models of real mechanical systems possible. In particular, research efforts have been focused on the efficient evaluation of the system Jacobians which have a crucial effect on the stability and accuracy of the linearly implicit Euler method. Classical off-line techniques based on finite-differences approximations are not suitable for RT simulations of large scale models due to the high number of function evaluations they require whenever the Jacobians must be updated. Moreover, in order to guarantee the unconditional stability of the linearly implicit Euler method, only the terms related to the stiff force elements in the vector of generalized forces are required, and thus the exact computation of Jacobians is not needed at all. The analytical evaluation of the Jacobians at each time step has been proposed by Rill in [6] where the choice of the stiff terms to be retained is based on physical considerations on the mechanical system. An alternative method has been proposed by Schiela et al. in [7] where, in a pre-processing step, a reference trajectory is selected for the evaluation of the Jacobians whose non-stiff terms are detected by means of numerical considerations and then brought to zero in order to enhance the factorization of the Jacobian matrix. A third approach has been used by Arnold et al. in [4] relying on a pre-processing step during which the Jacobians are evaluated for all the characteristic configurations of the mechanical system. This set of pre-evaluated Jacobians is then used during the RT simulation avoiding any additional computational effort.

In the present paper, we investigate the possible efficiency improvements in the linearly implicit Euler method associated to an automatic switching from a dependent to an independent coordinates' formulation. This approach can potentially improve the integration efficiency if it implies a substantial reduction in the number of degrees of freedom (DOFs) retained in the multibody model. This is particularly the case for multibody systems composed by a large number of bodies whose configuration parameters are related by a large number of constraint equations. Detailed multibody models developed for automotive applications perfectly fit in this category since they are generally built using a considerable number of bodies (more than 150) and ideal joints (e.g. driveline and steering systems' connections). Moreover, these models are usually obtained using general purpose multibody software based on dependent coordinates' formulations in which 4 dependent Euler parameters are employed to describe the orientation of each body. The use of dependent Euler parameters avoids numerical problems due to singular configurations but requires an additional constraint equation for each body in the system, causing a great increase in the total number of constraint equations. The proposed approach intends exploiting these intrinsic characteristics to improve the efficiency of the standard implementation of the linearly implicit Euler method for automotive applications. In particular, we modify the implementation of the linearly implicit Euler method in order to carry out the integration process by only considering a reduced set of independent velocities instead of the complete one. To this aim we apply the projection method proposed by the Jalón et al. in [8]. By expressing the EOMs of the multibody system in independent coordinates it is possible to obtain a great reduction in the dimension of the linear system which has to be solved at each time step, since only the current independent accelerations must be found instead of the complete set of dependent

accelerations. The proposed methodology has been implemented starting from a symbolic representation of the EOMs of the mechanical system. Analytical expressions for the Jacobians are obtained by considering only the stiff terms in the vector of generalized applied forces, and the non-iterative projection method proposed in [4] is applied as well in order to minimize the drift-off of the numerical solution from the constraint conditions. The proposed implementation of the linearly implicit Euler method is then compared to the classical one based on a dependent coordinates' formulation through a numerical test case in order to highlight possible advantages and drawbacks. The paper is organized as follows. The dependent coordinates' formulation of the EOMs of a multibody system is discussed in Sect. 2 together with the main features of implicit integration schemes. The linearly implicit Euler method for the integration of the EOMs expressed in dependent coordinates is addressed in Sect. 3. The algorithm adopted for an automated switching from dependent to independent coordinates within the linearly implicit Euler method is described in Sect. 4. Section 5 deals with the non-iterative projection step adopted to stabilize the constraint equations, and finally, in Sect. 6, the performances of the linearly implicit Euler method with a dependent and an independent formulation are compared.

2 Dependent coordinates formulation of the EOMs

Multibody models are composed by several bodies whose positions and orientations must be univocally described by means of a set of configuration parameters or coordinates [9]. Formulations in dependent coordinates constitute the basis for several general purpose multibody software. When dependent coordinates are used, the position and the orientation of each body in the mechanical system are described by means of a fixed number of configuration parameters while a set of algebraic equations is defined in order to mathematically represent the constraint conditions applied among bodies. This leads to the following system of index-3 DAE:

$$\begin{aligned} \mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \boldsymbol{\lambda}, t) &= \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \boldsymbol{\Phi}_q^T(\mathbf{q}, t)\boldsymbol{\lambda} - \mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}, t) = \mathbf{0}, \\ \boldsymbol{\Phi}(\mathbf{q}, t) &= \mathbf{0}. \end{aligned} \quad (1)$$

In Eq. (1), \mathbf{M} and \mathbf{Q} represent respectively the mass matrix and the vector of generalized applied forces. The latter contains the gravity force and the force elements applied between bodies, which in vehicle dynamics applications generally represent tires and shock-absorbers as well as elastic bushing connections and aerodynamics loads. The n dependent configuration parameters univocally describing the configuration of the multibody system and their time derivatives are indicated respectively as \mathbf{q} and $\dot{\mathbf{q}}$. Vector $\boldsymbol{\Phi}$ contains the constraint equations whose number is denoted as m , the constraint Jacobian matrix $\boldsymbol{\Phi}_q$ groups the derivatives of the constraint equations with respect to the configuration parameters, and finally, $\boldsymbol{\lambda}$ is the vector of Lagrange multipliers. Due to the modelling of bushing connections and shock absorbers' bump and rebound stops, the vector of generalized applied forces in Eq. (1) contains stiff terms which lead to a stiff index-3 DAE system whose direct integration cannot be approached by means of standard ODEs integrators. For this reason the index of the DAE system is generally reduced to 2 and 1 by substituting the constraint equations at the position level in Eq. (1) respectively with the constraints at the velocity coordinates' level,

$$\boldsymbol{\Phi}_q \dot{\mathbf{q}} + \boldsymbol{\Phi}_t = \mathbf{0}, \quad (2)$$

and by the constraints at the acceleration coordinates' level,

$$\Phi_q \ddot{\mathbf{q}} + \dot{\Phi}_t + \dot{\Phi}_q \dot{\mathbf{q}} = \mathbf{0}. \tag{3}$$

In Eqs. (2) and (3), the vector Φ_t contains the derivative of the constraint equations w.r.t. time, while $\dot{\Phi}_q$ is the derivative w.r.t. time of the constraint Jacobian matrix. Since the differentiation process cuts out the constant terms present in the constraint equations at the position level, a linear and quadratic growth of $\|\Phi(\mathbf{q}, t)\|$ can be observed in the numerical solutions obtained respectively with the *index-2* and *index-1* formulations. In order to avoid this drift away from the manifold defined by the constraint equations, the Baumgarte stabilization technique is commonly employed [10]. In the classical Baumgarte approach, the violation in the constraint conditions during the time integration of the EOMs expressed in the *index-1* form is reduced by substituting the constraint at the acceleration level of Eq. (3) by a proper linear combination of the constraint conditions at position, velocity and acceleration level:

$$\Phi_q \ddot{\mathbf{q}} + \dot{\Phi}_t + \dot{\Phi}_q \dot{\mathbf{q}} + 2\alpha(\Phi_q \dot{\mathbf{q}} + \Phi_t) + \beta^2 \Phi = \mathbf{0}. \tag{4}$$

The same approach can be used to apply the Baumgarte stabilization to the time integration of the EOMs expressed in the *index-2* form [4] by replacing Eq. (2) with

$$\Phi_q \dot{\mathbf{q}} + \Phi_t + \gamma \Phi = \mathbf{0}. \tag{5}$$

Proper numerical values of the Baumgarte parameters α , β and γ must be assigned in order to reduce the drift-off effect and to limit the introduction of artificial stiffness into the system.

The numerical integration of the EOMs expressed in the *index-1* and *index-2* forms is typically carried out by means of implicit integration schemes in off-line simulations. Indeed the presence of stiff terms in the EOMs imposes unpractical restrictions to the step-size if explicit integration schemes are used. In order to find the set of $3n + m$ unknowns constituted by positions, velocities, accelerations and Lagrange multipliers at the next time step, $2n$ additional integrals must be added to the EOMs of Eq. (1),

$$\mathbf{q}_{(i+1)} = \mathbf{q}_{(i)} + \int_{t_{(i)}}^{t_{(i)}+h} \dot{\mathbf{q}} dt, \quad \dot{\mathbf{q}}_{(i+1)} = \dot{\mathbf{q}}_{(i)} + \int_{t_{(i)}}^{t_{(i)}+h} \ddot{\mathbf{q}} dt, \tag{6}$$

where h is the time-step size and the subscripts i and $i + 1$ indicate respectively the current and the next time step of the integration process. Implicit integration algorithms rely on the substitution of the integrals of Eq. (6) for integration formulas which assume the following form:

$$\mathbf{q}_{(i+1)} = \mathbf{\Lambda}(\mathbf{q}_{\leq i}, \dot{\mathbf{q}}_{\leq i}, \ddot{\mathbf{q}}_{\leq i}, \ddot{\mathbf{q}}_{(i+1)}), \quad \dot{\mathbf{q}}_{(i+1)} = \dot{\mathbf{\Lambda}}(\dot{\mathbf{q}}_{\leq i}, \ddot{\mathbf{q}}_{\leq i}, \ddot{\mathbf{q}}_{(i+1)}). \tag{7}$$

In Eq. (7), the dependent configuration parameters and the dependent velocities at the next time step are expressed as functions of the states at the current and previous time steps as well as functions of the accelerations at the next time step. The actual form of the integration formulas $\mathbf{\Lambda}$ and $\dot{\mathbf{\Lambda}}$ varies according to the particular implicit scheme adopted. The configuration parameters \mathbf{q} and their time derivatives $\dot{\mathbf{q}}$ in the EOMs expressed both in the *index-1* or *index-2* formulation can be replaced with the integration formulas of Eq. (7) in order to obtain a nonlinear system having as unknowns the n accelerations and the m Lagrange multipliers at the next time step,

$$\mathbf{F}(\ddot{\mathbf{q}}_{(i+1)}, \lambda_{(i+1)}) = \begin{bmatrix} \mathbf{f}(\ddot{\mathbf{q}}_{(i+1)}, \lambda_{(i+1)}) \\ \mathbf{CEs}(\ddot{\mathbf{q}}_{(i+1)}) \end{bmatrix} = \mathbf{0}, \tag{8}$$

where $\mathbf{CEs}(\dot{\mathbf{q}}_{(i+1)})$ indicates the constraint equations (at the velocity or acceleration level). The nonlinear system in Eq. (8) can be solved iteratively by the Newton–Raphson algorithm in order to find the accelerations and the Lagrange multipliers at the next time step. The k th iteration of the Newton–Raphson method can be written as

$$\begin{Bmatrix} \ddot{\mathbf{q}}_{(i+1)}^k \\ \boldsymbol{\lambda}_{(i+1)}^k \end{Bmatrix} = \begin{Bmatrix} \ddot{\mathbf{q}}_{(i+1)}^{k-1} \\ \boldsymbol{\lambda}_{(i+1)}^{k-1} \end{Bmatrix} - [\mathbf{J}(\dot{\mathbf{q}}_{(i+1)}^{k-1}, \boldsymbol{\lambda}_{(i+1)}^{k-1})]^{-1} \mathbf{F}(\dot{\mathbf{q}}_{(i+1)}^{k-1}, \boldsymbol{\lambda}_{(i+1)}^{k-1}) \tag{9}$$

where the iteration matrix \mathbf{J} is defined as

$$\mathbf{J} = \begin{bmatrix} \frac{\partial \mathbf{f}(\dot{\mathbf{q}}_{(i+1)}, \boldsymbol{\lambda}_{(i+1)})}{\partial \dot{\mathbf{q}}_{(i+1)}} & \frac{\partial \mathbf{f}(\dot{\mathbf{q}}_{(i+1)}, \boldsymbol{\lambda}_{(i+1)})}{\partial \boldsymbol{\lambda}_{(i+1)}} \\ \frac{\partial \mathbf{CEs}(\dot{\mathbf{q}}_{(i+1)})}{\partial \dot{\mathbf{q}}_{(i+1)}} & \mathbf{0} \end{bmatrix}. \tag{10}$$

As previously pointed out, the basic requirement in RT simulations is that the number of operations to be carried out at each integration time step must be fixed. Indeed, this is the only way to be a priori sure that the turnaround time, i.e. the time required to complete the operations at each time step, is lower than the time span of 1 ms imposed by RT standards [4, 7]. Due to their iterative nature, implicit integration schemes are not able to fulfill this requirement. For this reason the linearly implicit Euler integration method has been proposed [4–6], which is able to handle the integration of DAE systems containing stiff terms by performing a fixed number of operations at each time step. This is achieved by means of a linear approximation of the stiff terms contained in the vector of the generalized applied forces \mathbf{Q} in Eq. (1) as it will be explained in the next section.

3 Integration of the EOMs expressed in dependent coordinates with the linearly implicit Euler method

The formulas of the first order explicit Euler method,

$$\begin{bmatrix} \mathbf{q}_{(i+1)} \\ \dot{\mathbf{q}}_{(i+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{q}_{(i)} \\ \dot{\mathbf{q}}_{(i)} \end{bmatrix} + h \begin{bmatrix} \dot{\mathbf{q}}_{(i)} \\ \ddot{\mathbf{q}}_{(i)} \end{bmatrix}, \tag{11}$$

can be applied for integrating the EOMs of a multibody system represented in dependent coordinates by obtaining the acceleration $\ddot{\mathbf{q}}_{(i)}$ at the current time step i as the solution of the following linear system:

$$\begin{bmatrix} \mathbf{M}_{(i)} & \boldsymbol{\Phi}_{\mathbf{q}_{(i)}}^T \\ \boldsymbol{\Phi}_{\mathbf{q}_{(i)}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}}_{(i)} \\ \boldsymbol{\lambda}_{(i)} \end{bmatrix} = \begin{bmatrix} \mathbf{Q}_{(i)} \\ -\dot{\boldsymbol{\Phi}}_{\mathbf{q}_{(i)}} \dot{\mathbf{q}}_{(i)} - \dot{\boldsymbol{\Phi}}_{t_{(i)}} \end{bmatrix} \tag{12}$$

where $\mathbf{M}_{(i)} = \mathbf{M}(\mathbf{q}_{(i)})$, $\mathbf{Q}_{(i)} = \mathbf{Q}(\mathbf{q}_{(i)}, \dot{\mathbf{q}}_{(i)}, t_{(i)})$, $\dot{\boldsymbol{\Phi}}_{\mathbf{q}_{(i)}} = \dot{\boldsymbol{\Phi}}_{\mathbf{q}}(\mathbf{q}_{(i)}, t_{(i)})$ and $\dot{\boldsymbol{\Phi}}_{t_{(i)}} = \dot{\boldsymbol{\Phi}}_t(\mathbf{q}_{(i)}, t_{(i)})$. Equation (12) has been obtained by discretizing in time the differential equations of motion in Eq. (1) and the constraints at the accelerations level in Eq. (3) and by expressing them at the current time step i . The variations in the values of the dependent configuration parameters $\Delta \mathbf{q}_{(i)}$ and their time derivatives $\Delta \dot{\mathbf{q}}_{(i)}$ are defined by the following forward differences formulas:

$$\begin{bmatrix} \Delta \mathbf{q}_{(i)} \\ \Delta \dot{\mathbf{q}}_{(i)} \end{bmatrix} = \begin{bmatrix} \mathbf{q}_{(i+1)} - \mathbf{q}_{(i)} \\ \dot{\mathbf{q}}_{(i+1)} - \dot{\mathbf{q}}_{(i)} \end{bmatrix}. \tag{13}$$

By introducing Eqs. (12) and (13) into Eq. (11), the following matrix expression for the explicit Euler method can be obtained:

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}^{(i)} \end{bmatrix} \begin{bmatrix} \Delta \mathbf{q}^{(i)} \\ \Delta \dot{\mathbf{q}}^{(i)} \end{bmatrix} = h \begin{bmatrix} \dot{\mathbf{q}}^{(i)} \\ -\Phi_{\mathbf{q}^{(i)}}^T \boldsymbol{\lambda}^{(i)} + \mathbf{Q}^{(i)} \end{bmatrix}. \tag{14}$$

It is well known that the explicit Euler method cannot be used for the stable integration of stiff ODEs or DAEs systems unless the time step is reduced to unpractical values [11]. An alternative is the use of the *implicit Euler* method which can handle the integration of ODEs or DAEs systems containing stiff terms. The formulas of the implicit Euler method are:

$$\begin{bmatrix} \mathbf{q}^{(i+1)} \\ \dot{\mathbf{q}}^{(i+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{q}^{(i)} \\ \dot{\mathbf{q}}^{(i)} \end{bmatrix} + h \begin{bmatrix} \dot{\mathbf{q}}^{(i+1)} \\ \ddot{\mathbf{q}}^{(i+1)} \end{bmatrix} \tag{15}$$

where the accelerations $\ddot{\mathbf{q}}^{(i+1)}$ at the next time step $i + 1$ are given by the solution of the following system:

$$\begin{bmatrix} \mathbf{M}^{(i+1)} & \Phi_{\mathbf{q}^{(i+1)}}^T \\ \Phi_{\mathbf{q}^{(i+1)}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}}^{(i+1)} \\ \boldsymbol{\lambda}^{(i+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{Q}^{(i+1)} \\ -\dot{\Phi}_{\mathbf{q}^{(i+1)}} \dot{\mathbf{q}}^{(i+1)} - \dot{\Phi}_{t^{(i+1)}} \end{bmatrix} \tag{16}$$

where $\mathbf{M}^{(i+1)} = \mathbf{M}(\mathbf{q}^{(i+1)})$, $\mathbf{Q}^{(i+1)} = \mathbf{Q}(\mathbf{q}^{(i+1)}, \dot{\mathbf{q}}^{(i+1)}, t^{(i+1)})$, $\dot{\Phi}_{\mathbf{q}^{(i+1)}} = \dot{\Phi}_{\mathbf{q}}(\mathbf{q}^{(i+1)}, t^{(i+1)})$ and $\dot{\Phi}_{t^{(i+1)}} = \dot{\Phi}_t(\mathbf{q}^{(i+1)}, t^{(i+1)})$. The matrix expression for the implicit Euler method can be obtained by introducing Eqs. (16) and (13) into Eq. (15):

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}^{(i+1)} \end{bmatrix} \begin{bmatrix} \Delta \mathbf{q}^{(i)} \\ \Delta \dot{\mathbf{q}}^{(i)} \end{bmatrix} = h \begin{bmatrix} \dot{\mathbf{q}}^{(i+1)} \\ -\dot{\Phi}_{\mathbf{q}^{(i+1)}}^T \boldsymbol{\lambda}^{(i+1)} + \mathbf{Q}^{(i+1)} \end{bmatrix}. \tag{17}$$

The solution of the nonlinear system in Eq. (17) requires an a priori unknown number of iterations to find the variations in the dependent positions and velocities at the current time step.

In order to avoid the step-size restrictions related to the explicit Euler integration scheme and the iterative solution of a nonlinear system of equations associated to the implicit Euler integration scheme, the *linearly implicit Euler method* has been proposed [4, 5] for the real-time integration of the stiff DAE system of Eq. (1). The linearly implicit Euler method represents a suitable alternative to the explicit and the implicit Euler methods. Indeed, even if it requires a fixed number of operations for each time-step, it can handle the integration of stiff EOMs by exploiting a linear approximation of the stiff terms in the vector of generalized applied forces. In particular, the matrix formulation of the linearly implicit Euler method can be obtained from the explicit Euler's one by taking into account the Jacobian of the RHS in Eq. (14) [4, 11]:

$$\left(\begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}^{(i)} \end{bmatrix} - h \mathbf{J}^{(i)} \right) \begin{bmatrix} \Delta \mathbf{q}^{(i)} \\ \Delta \dot{\mathbf{q}}^{(i)} \end{bmatrix} = h \begin{bmatrix} \dot{\mathbf{q}}^{(i)} \\ -\Phi_{\mathbf{q}^{(i)}}^T \boldsymbol{\lambda}^{(i)} + \mathbf{Q}^{(i)} \end{bmatrix}. \tag{18}$$

Since the stiffness in the multibody system is mainly due to the stiff terms in the vector of generalized applied forces, the Jacobian matrix $\mathbf{J}^{(i)}$ does not include the terms related to the constraints reactions and is defined as

$$\mathbf{J}^{(i)} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{J}_{\mathbf{q}^{(i)}} & \mathbf{J}_{\dot{\mathbf{q}}^{(i)}} \end{bmatrix} \tag{19}$$

Table 1 Algorithm 1 and corresponding numerical integration routine

algorithm_1 $[\Delta \mathbf{q}_{(i)} \ \Delta \dot{\mathbf{q}}_{(i)}] = \text{Function_Evaluation_I}(\mathbf{q}_{(i)}, \dot{\mathbf{q}}_{(i)}, t_{(i)})$

$\mathbf{M}_{(i)} = \mathbf{M}(\mathbf{q}_{(i)})$

$\mathbf{Q}_{(i)} = \mathbf{Q}(\mathbf{q}_{(i)}, \dot{\mathbf{q}}_{(i)}, t_{(i)})$

$\Phi_{\mathbf{q}_{(i)}} = \Phi_{\mathbf{q}}(\mathbf{q}_{(i)}, t_{(i)})$

$\mathbf{J}_{\mathbf{q}_{(i)}} = \mathbf{J}_{\mathbf{q}_{(i)}}(\mathbf{q}_{(i)}, \dot{\mathbf{q}}_{(i)})$

$\mathbf{J}_{\dot{\mathbf{q}}_{(i)}} = \mathbf{J}_{\dot{\mathbf{q}}_{(i)}}(\mathbf{q}_{(i)}, \dot{\mathbf{q}}_{(i)})$

$\Phi_{\mathbf{q}_{(i+1)}} = \Phi_{\mathbf{q}}(\mathbf{q}_{(i)} + \Delta \mathbf{q}_{(i)}, t_{(i+1)})$

$\Phi_{t_{(i+1)}} = \Phi_t(\mathbf{q}_{(i)} + \Delta \mathbf{q}_{(i)}, t_{(i+1)})$

$$[\Delta \dot{\mathbf{q}}_{(i)} \ \lambda_{(i)}]^T = \begin{bmatrix} \mathbf{M}_{(i)} - h\mathbf{J}_{\dot{\mathbf{q}}_{(i)}} - h^2\mathbf{J}_{\mathbf{q}_{(i)}} & h\Phi_{\dot{\mathbf{q}}_{(i)}}^T \\ \Phi_{\mathbf{q}_{(i+1)}} & \mathbf{0} \end{bmatrix}^{-1} \begin{bmatrix} h\mathbf{Q}_{(i)} + h^2\mathbf{J}_{\mathbf{q}_{(i)}}\dot{\mathbf{q}}_{(i)} \\ -\Phi_{\mathbf{q}_{(i+1)}}\dot{\mathbf{q}}_{(i)} - \Phi_{t_{(i+1)}} \end{bmatrix}$$

Numerical Integration 1 $[\mathbf{q}, \dot{\mathbf{q}}] = \text{Num_Int_I}(\mathbf{q}_{(0)}, \dot{\mathbf{q}}_{(0)})$

for $i = 1$ to i_{end}

$[\Delta \mathbf{q}_{(i)} \ \Delta \dot{\mathbf{q}}_{(i)}] = \text{Function_Evaluation_I}(\mathbf{q}_{(i)}, \dot{\mathbf{q}}_{(i)}, t_{(i)})$

$$\begin{bmatrix} \mathbf{q}_{(i+1)} \\ \dot{\mathbf{q}}_{(i+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{q}_{(i)} \\ \dot{\mathbf{q}}_{(i)} \end{bmatrix} + \begin{bmatrix} \Delta \mathbf{q}_{(i)} \\ \Delta \dot{\mathbf{q}}_{(i)} \end{bmatrix}$$

where the Jacobians of the vector of generalized applied forces are indicated as $\mathbf{J}_{\mathbf{q}_{(i)}} = \frac{\partial \mathbf{Q}}{\partial \mathbf{q}}|_{t_{(i)}}$ and $\mathbf{J}_{\dot{\mathbf{q}}_{(i)}} = \frac{\partial \mathbf{Q}}{\partial \dot{\mathbf{q}}}|_{t_{(i)}}$.

An *index-2* DAE formulation of the EOMs can greatly reduce the drift off of the numerical solution from the manifold defined by the algebraic constraint equations. As suggested by Arnold et al. in [4], this can be achieved by discretizing in time Eq. (2) and by imposing that the velocities at the next time step satisfy the constraints at the velocity level,

$$\Phi_{\mathbf{q}}(\mathbf{q}_{(i+1)}, t_{(i+1)})\dot{\mathbf{q}}_{(i+1)} + \Phi_t(\mathbf{q}_{(i+1)}, t_{(i+1)}) = \mathbf{0}. \tag{20}$$

The expressions for the positions and velocities at the next time step of Eq. (13) can be then substituted into Eq. (20) to obtain

$$\Phi_{\mathbf{q}_{(i+1)}}[\dot{\mathbf{q}}_{(i)} + \Delta \dot{\mathbf{q}}_{(i)}] + \Phi_{t_{(i+1)}} = \mathbf{0}. \tag{21}$$

The m additional relations of Eq. (21) can finally be combined with Eq. (18) in order to obtain the following system of linear equations which can be solved to find the variations in the n dependent velocities and the m Lagrange multipliers at the current time step:

$$\begin{bmatrix} \mathbf{M}_{(i)} - h\mathbf{J}_{\dot{\mathbf{q}}_{(i)}} - h^2\mathbf{J}_{\mathbf{q}_{(i)}} & h\Phi_{\dot{\mathbf{q}}_{(i)}}^T \\ \Phi_{\mathbf{q}_{(i+1)}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \Delta \dot{\mathbf{q}}_{(i)} \\ \lambda_{(i)} \end{bmatrix} = \begin{bmatrix} h\mathbf{Q}_{(i)} + h^2\mathbf{J}_{\mathbf{q}_{(i)}}\dot{\mathbf{q}}_{(i)} \\ -\Phi_{\mathbf{q}_{(i+1)}}\dot{\mathbf{q}}_{(i)} - \Phi_{t_{(i+1)}} \end{bmatrix}. \tag{22}$$

Equation (22) provides the function evaluation associated with the linearly implicit Euler method for a constrained multibody system whose EOMs are expressed in terms of dependent coordinates. For linear systems, the unconditional stability of the integration method is guaranteed as long as the matrices $\mathbf{J}_{\mathbf{q}_{(i)}}$ and $\mathbf{J}_{\dot{\mathbf{q}}_{(i)}}$ approximate all the stiff terms in the exact Jacobians $\frac{\partial \mathbf{Q}}{\partial \mathbf{q}}|_{t_{(i)}}$ and $\frac{\partial \mathbf{Q}}{\partial \dot{\mathbf{q}}}|_{t_{(i)}}$ [5].

The pseudo-code implemented to perform the function evaluation of Eq. (22) is reported in Table 1 together with the pseudo-code associated to the numerical integration of the EOMs. It can be noticed that the number of operations required to perform the function evaluation at each time step is fixed, thus meeting the basic requirement for RT simulations.

4 Automated independent coordinates' switching in the linearly implicit Euler method

The linearly implicit Euler method presented in the previous section is able to handle the integration of the stiff DAE system of Eq. (1) by performing a fixed number of operations at each time step. With the aim of improving the numerical efficiency of this method, we investigate the use of the projection method based on the matrix \mathbf{R} defining the basis of the null-space of the constraint Jacobian, in order to reduce the dimension of the linear system to be solved at each time step. In particular, we propose the use of the projection method at each iteration of the linearly implicit Euler method to automatically transform the expressions for the function evaluation of Eq. (22) from a dependent coordinates' formulation to a state space form in terms of a minimal set of independent coordinates, allowing a reduction in the dimension of the linear system from $n + m$ to $f = n - m$. This reduction in the number of unknowns can be particularly important if a highly constrained mechanical system is taken into account. For tree-structured multibody systems, the use of relative joints' coordinates represents an obvious choice if a set of independent coordinates must be defined. On the contrary, for closed-loop systems such as those employed in automotive applications, the definition of a proper set of independent coordinates is not straightforward and requires the employment of tailored numerical techniques [8]. Among them the projection method based on the matrix \mathbf{R} relies on the relationships between independent and dependent velocities and accelerations [8]. In order to obtain these relationships, the vectors \mathbf{b} and \mathbf{c} must be defined from Eqs. (2) and (3) as

$$\begin{aligned} \Phi_q \dot{\mathbf{q}} &= -\dot{\Phi}_t \equiv \mathbf{b}, \\ \Phi_q \ddot{\mathbf{q}} &= -\ddot{\Phi}_t - \dot{\Phi}_q \dot{\mathbf{q}} \equiv \mathbf{c}. \end{aligned} \tag{23}$$

In addition to Eq. (23), it must be also considered that the independent velocities $\dot{\mathbf{z}}$ can be obtained as the projection of the dependent velocities on the rows of a matrix \mathbf{B} of dimensions $f \times n$. The set of independent velocities is determined by the choice of matrix \mathbf{B} . A particular set of independent coordinates is not adequate to univocally determine the configuration of a mechanism during its entire range of motion due to possible singular configurations. For this reason matrix \mathbf{B} must be changed whenever the configuration of the mechanism cannot be further described by means of the current set of independent coordinates. However, for an automotive application as addressed in this paper, matrix \mathbf{B} can be considered constant since no singular configurations are reached during normal working conditions. Different numerical methods can be employed in order to obtain matrix \mathbf{B} starting from the knowledge of the constraint Jacobian matrix [8, 12]. In particular, matrix \mathbf{B} must have full rank f and its rows must be also linearly independent of the m rows of the constraint Jacobian matrix. Under these assumptions, the relationships between independent and dependent velocities and accelerations can be defined respectively as [8]:

$$\dot{\mathbf{q}} = \begin{bmatrix} \Phi_q \\ \mathbf{B} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{b} \\ \dot{\mathbf{z}} \end{bmatrix} = [\mathbf{S} \quad \mathbf{R}] \begin{bmatrix} \mathbf{b} \\ \dot{\mathbf{z}} \end{bmatrix} = \mathbf{S}\mathbf{b} + \mathbf{R}\dot{\mathbf{z}}, \tag{24}$$

$$\ddot{\mathbf{q}} = \begin{bmatrix} \Phi_q \\ \mathbf{B} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{c} \\ \ddot{\mathbf{z}} \end{bmatrix} = [\mathbf{S} \quad \mathbf{R}] \begin{bmatrix} \mathbf{c} \\ \ddot{\mathbf{z}} \end{bmatrix} = \mathbf{S}\mathbf{c} + \mathbf{R}\ddot{\mathbf{z}}. \tag{25}$$

Equations (24) and (25) can be rewritten in a discretized form as:

$$\begin{aligned}\Delta \mathbf{q}_{(i)} &= h\mathbf{S}_{(i)}\mathbf{b}_{(i)} + \mathbf{R}_{(i)}\Delta \mathbf{z}_{(i)}, \\ \Delta \dot{\mathbf{q}}_{(i)} &= h\mathbf{S}_{(i)}\mathbf{c}_{(i)} + \mathbf{R}_{(i)}\Delta \dot{\mathbf{z}}_{(i)}.\end{aligned}\quad (26)$$

The columns of the $n \times f$ matrix \mathbf{R} in Eqs. (24), (25) and (26) constitute a basis of the null-space of the constraint Jacobian matrix, i.e. $\Phi_{\mathbf{q}}\mathbf{R} = \mathbf{0}$, as can be verified by considering the following matrix expression [8] resulting from the definition of matrix \mathbf{R} given in Eqs. (24) and (25):

$$\begin{bmatrix} \Phi_{\mathbf{q}} \\ \mathbf{B} \end{bmatrix} \begin{bmatrix} \Phi_{\mathbf{q}} \\ \mathbf{B} \end{bmatrix}^{-1} = \begin{bmatrix} \Phi_{\mathbf{q}} \\ \mathbf{B} \end{bmatrix} [\mathbf{S} \quad \mathbf{R}] = \begin{bmatrix} \Phi_{\mathbf{q}}\mathbf{S} & \Phi_{\mathbf{q}}\mathbf{R} \\ \mathbf{B}\mathbf{S} & \mathbf{B}\mathbf{R} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}. \quad (27)$$

This property can be exploited in order to eliminate the term containing the Lagrange multipliers in Eq. (22). Indeed, by substituting the second one of Eq. (26) into the first equation of the matrix expression in Eq. (22) and pre-multiplying by the transpose of the projection matrix \mathbf{R} , one obtains

$$\begin{aligned}\mathbf{R}_{(i)}^T [\mathbf{M}_{(i)} - h\mathbf{J}_{\dot{\mathbf{q}}_{(i)}} - h^2\mathbf{J}_{\mathbf{q}_{(i)}}] \mathbf{R}_{(i)} \Delta \dot{\mathbf{z}}_{(i)} \\ = \mathbf{R}_{(i)}^T [h\mathbf{Q}_{(i)} + h^2\mathbf{J}_{\mathbf{q}_{(i)}}\dot{\mathbf{q}}_{(i)}] - \mathbf{R}_{(i)}^T [\mathbf{M}_{(i)} - h\mathbf{J}_{\dot{\mathbf{q}}_{(i)}} - h^2\mathbf{J}_{\mathbf{q}_{(i)}}] h\mathbf{S}_{(i)}\mathbf{c}_{(i)}.\end{aligned}\quad (28)$$

The linear system of Eq. (28) has only f variations in the independent velocities as unknowns and represents the function evaluation needed by the linearly implicit Euler method at each time step. Two possible algorithms can be employed in order to carry on the integration process by using the state space formulation of Eq. (28) as suggested by de Jalón et al. in [8]. A first possibility is to employ the variations in the independent coordinates and velocities to find the independent coordinates and velocities at the next time step by means of an Euler integration step. The dependent velocities and positions at the next time step must then be found by solving the velocity and the position problems in order to be able to proceed with the integration process. Indeed, the dependent velocities and positions are required at each time step for the computation of the terms in Eq. (28). An alternative approach can be used [8] which allows avoiding the solution of the nonlinear position problem at each time step. This approach relies on the integration of an enlarged system of differential equations which practically translates in giving as input the current variations in the dependent coordinates $\Delta \mathbf{q}_{(i)}$ and in the independent velocities $\Delta \dot{\mathbf{z}}_{(i)}$ to the Euler integration step obtaining as output the dependent coordinates $\mathbf{q}_{(i)}$ and the independent velocities $\dot{\mathbf{z}}_{(i)}$ at the next time step.

The pseudo-code implementing the function evaluation proposed in this section is reported in Table 2. The routine receives as input the constant matrix \mathbf{B} , the dependent coordinates and the independent velocities at the current time step and returns as output the current variations in the dependent positions and in the independent velocities. The dimension of the linear system to be solved in order to find the variations in the independent velocities at the current time step is reduced to $f = n - m$ by transforming the EOMs from a dependent coordinates' formulation to a state space form in terms of a minimal set of coordinates. However, in order to find the projection matrix \mathbf{R} , a matrix of dimensions $n \times n$ must be inverted at each time step in Algorithm 2, reducing the advantages of the state space formulation. Notice that the constant matrix \mathbf{B} representing the mapping from dependent to independent velocities is computed just once before the beginning of the simulation using one of the methods proposed in [8, 12] and then given as input to Algorithm 2.

Table 2 Algorithm 2 and corresponding numerical integration routine

algorithm_2 $[\Delta \mathbf{q}_{(i)} \ \Delta \dot{\mathbf{z}}_{(i)}] = \text{Function_Evaluation_2}(\mathbf{q}_{(i)}, \dot{\mathbf{z}}_{(i)}, t_{(i)}, \mathbf{B})$

$\Phi_{\mathbf{q}_{(i)}} = \Phi_{\mathbf{q}}(\mathbf{q}_{(i)}, t_{(i)})$

$\Phi_{t_{(i)}} = \Phi_t(\mathbf{q}_{(i)}, t_{(i)})$

$\mathbf{T}_{(i)} = [\mathbf{S}_{(i)} \ \mathbf{R}_{(i)}] = \begin{bmatrix} \Phi_{\mathbf{q}_{(i)}} \\ \mathbf{B} \end{bmatrix}^{-1}$

$\dot{\mathbf{q}}_{(i)} = \mathbf{T}_{(i)} \begin{bmatrix} -\Phi_{t_{(i)}} \\ \dot{\mathbf{z}}_{(i)} \end{bmatrix}$

$\mathbf{M}_{(i)} = \mathbf{M}(\mathbf{q}_{(i)})$

$\mathbf{Q}_{(i)} = \mathbf{Q}(\mathbf{q}_{(i)}, \dot{\mathbf{q}}_{(i)}, t_{(i)})$

$\mathbf{J}_{\mathbf{q}_{(i)}} = \mathbf{J}_{\mathbf{q}_{(i)}}(\mathbf{q}_{(i)}, \dot{\mathbf{q}}_{(i)})$

$\mathbf{J}_{\dot{\mathbf{q}}_{(i)}} = \mathbf{J}_{\dot{\mathbf{q}}_{(i)}}(\mathbf{q}_{(i)}, \dot{\mathbf{q}}_{(i)})$

$\mathbf{c}_{(i)} = c(\mathbf{q}_{(i)}, \dot{\mathbf{q}}_{(i)}, t_{(i)})$

$\mathbf{S}_{(i)} \mathbf{c}_{(i)} = \mathbf{T}_{(i)} [\mathbf{c}_{(i)} \ \mathbf{0}]^T$

$\Delta \dot{\mathbf{z}}_{(i)} = [\mathbf{R}_{(i)}^T [\mathbf{M}_{(i)} - h \mathbf{J}_{\mathbf{q}_{(i)}} - h^2 \mathbf{J}_{\mathbf{q}_{(i)}}] \mathbf{R}_{(i)}]^{-1}$
 $\cdot [\mathbf{R}_{(i)}^T [h \mathbf{Q}_{(i)} + h^2 \mathbf{J}_{\mathbf{q}_{(i)}} \dot{\mathbf{q}}_{(i)}] - \mathbf{R}_{(i)}^T [\mathbf{M}_{(i)} - h \mathbf{J}_{\mathbf{q}_{(i)}} - h^2 \mathbf{J}_{\mathbf{q}_{(i)}}] h \mathbf{S}_{(i)} \mathbf{c}_{(i)}]$

Numerical Integration 2 $[\mathbf{q}, \dot{\mathbf{z}}] = \text{Num_Int_2}(\mathbf{q}_{(0)}, \dot{\mathbf{z}}_{(0)})$

for $i = 1$ to i_{end}

$[\Delta \mathbf{q}_{(i)} \ \Delta \dot{\mathbf{z}}_{(i)}] = \text{Function_Evaluation_2}(\mathbf{q}_{(i)}, \dot{\mathbf{z}}_{(i)}, t_{(i)}, \mathbf{B})$

$\begin{bmatrix} \mathbf{q}_{(i+1)} \\ \dot{\mathbf{z}}_{(i+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{q}_{(i)} \\ \dot{\mathbf{z}}_{(i)} \end{bmatrix} + \begin{bmatrix} \Delta \mathbf{q}_{(i)} \\ \Delta \dot{\mathbf{z}}_{(i)} \end{bmatrix}$

5 Stabilization of the constraint equations

The numerical solutions obtained by means of the two implementations of the linearly implicit Euler method described in the previous sections are affected by the drift-off from the manifold defined by the algebraic constraint equations $\Phi(\mathbf{q}_{(i)}, t_{(i)}) = \mathbf{0}$. In particular, when the EOMs in terms of dependent coordinates are expressed in the *index-2* form of Eq. (22), the constraints at the velocity level are used in order to provide the m additional equations required to compute the variations in the dependent velocities and the Lagrange multipliers at the current time step. As previously pointed out, since the expressions of the constraints at the velocity level do not contain the constant terms originally present in the constraint equations at the position level, a linear growth of $\|\Phi(\mathbf{q}, t)\|$ takes place as the numerical solution advances in time [11]. Similarly, when the EOMs expressed in the state space formulation of Eq. (28) are integrated to obtain the dependent positions and the independent velocities at the next time step, a drift-off of the solution from the constraint manifold is also observed. This drift is due to the fact that the algorithm avoids the computationally expensive solution of the nonlinear position problem defined by the algebraic constraint equations at each time step. Numerical round-off errors can thus accumulate as the numerical integration advances in time, causing a violation of the constraint conditions.

A projection of the numerical solution back to the manifold defined by the constraint equations must be thus applied at the end of both Algorithms 1 and 2 in order to maintain an acceptable level of accuracy. Several techniques have been proposed in order to prevent the numerical solution to drift off from the manifold defined by the algebraic constraint equations [4, 10, 11]. A typical approach adopted in off-line simulation is the use of iterative

projection methods [13]. Generally the norm of the residual vector in the constraint equations at the next time step is evaluated and, if it is higher than a user defined tolerance, a nonlinear constrained minimization problem is solved iteratively in order to find a new set of dependent configuration parameters $\mathbf{q}_{(i+1)}^*$ which belongs again to the constraint manifold as described in [4, 5, 13]:

$$\begin{aligned} &\text{if } \|\Phi(\mathbf{q}_{(i)} + \Delta\mathbf{q}_{(i)}, t_{(i+1)})\| > TOL \\ &\text{then } \min\{\|\mathbf{q}_{(i+1)}^* - [\mathbf{q}_{(i)} + \Delta\mathbf{q}_{(i)}]\| : \Phi(\mathbf{q}_{(i+1)}^*, t_{(i+1)}) = \mathbf{0}\}. \end{aligned} \quad (29)$$

After the iterative solution of the constrained minimization problem of Eq. (29), the errors in the velocity constraints are corrected as well by projecting the velocities back to the manifold defined by the constraints at the velocity level [4, 5]. Once again the iterative solution of the minimization problem in Eq. (29) must be avoided in RT applications where an a priori known number of operations is required at each time step. As demonstrated by Burgermeister et al. [5], one Newton iteration for the solution of the constrained minimization problem mentioned above is sufficient to avoid the drift-off effect, provided that the projection is performed at each time step. The non-iterative projection method can be applied after the function evaluation in both Algorithms 1 and 2 in order to obtain a corrected set of variations in the dependent coordinates $\Delta\mathbf{q}_{(i)}^*$ from the solution of the following linear system [4]:

$$\begin{bmatrix} \mathbf{M}_{(i)} & h\Phi_{\mathbf{q}_{(i)}}^T \\ \Phi_{\mathbf{q}_{(i)}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{q}_{(i)}^* - \Delta\mathbf{q}_{(i)} \\ \boldsymbol{\mu}_{(i)} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ -\Phi(\mathbf{q}_{(i)} + \Delta\mathbf{q}_{(i)}, t_{(i+1)}) \end{bmatrix}. \quad (30)$$

As described in [4, 5], the linear system of Eq. (30) can be interpreted as one simplified step of the Newton–Raphson algorithm for the solution of the constrained minimization problem of Eq. (29) with the initial guess $\Delta\mathbf{q}_{(i)}^* = h\dot{\mathbf{q}}_{(i)}$ and $\boldsymbol{\mu}_{(i)} = \mathbf{0}$. The non-iterative projection step in Eq. (30) must be applied after the function evaluations of both Algorithms 1 and 2 to obtain the corrected variations in the dependent coordinates $\Delta\mathbf{q}_{(i)}^*$ which can then be used within the integration routines to carry on with the integration process as reported in the pseudo-codes of Table 3. The use of the projection step of Eq. (30) is sufficient to drastically reduce the drift away of the numerical solution from the constraint conditions [4, 5].

6 Numerical benchmark

In order to test and compare the efficiency and the accuracy of the two implementations of the linearly implicit Euler method presented in Sects. 2 and 3, an industrial application case has been assessed. The main goal is to highlight possible advantages of the implementation proposed in this paper, which exploits the projection method based on matrix \mathbf{R} to automatically convert the EOMs from a dependent to an independent coordinates' representation.

6.1 Industrial application case

A rear left multilink suspension of a rear-drive passenger car has been considered as shown in Fig. 1. The mechanical system is composed of 11 bodies: the chassis, the rear subframe, the 5 suspension's links, the knuckle, the rim, the differential shaft, and the halfshaft. The chassis is constrained in such a way that only its 3 translational DOFs are permitted and it is connected to the rear subframe by means of 4 bushings with linear stiffness and damping properties. The driving torque is delivered from the chassis to the rim through the driveline

Table 3 Numerical integration routines plus the non-iterative projection step

Numerical Integration 1 $[\mathbf{q}, \dot{\mathbf{q}}] = \text{Num_Int_1}(\mathbf{q}(0), \dot{\mathbf{q}}(0))$

for $i = 1$ to i_{end}

$[\Delta \mathbf{q}_{(i)} \ \Delta \dot{\mathbf{q}}_{(i)}] = \text{Function_Evaluation_1}(\mathbf{q}_{(i)}, \dot{\mathbf{q}}_{(i)}, t_{(i)})$

$$\begin{bmatrix} \Delta \mathbf{q}_{(i)}^* - \Delta \mathbf{q}_{(i)} \\ \boldsymbol{\mu}_{(i)} \end{bmatrix} = \begin{bmatrix} \mathbf{M}_{(i)} & h \boldsymbol{\Phi}_{(i)}^T \\ \boldsymbol{\Phi}_{(i)} & \mathbf{0} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{0} \\ -\boldsymbol{\Phi}(\mathbf{q}_{(i)} + \Delta \mathbf{q}_{(i)}, t_{(i+1)}) \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{q}_{(i+1)} \\ \dot{\mathbf{q}}_{(i+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{q}_{(i)} \\ \dot{\mathbf{q}}_{(i)} \end{bmatrix} + \begin{bmatrix} \Delta \mathbf{q}_{(i)}^* \\ \Delta \dot{\mathbf{q}}_{(i)} \end{bmatrix}$$

Numerical Integration 2 $[\mathbf{q}, \dot{\mathbf{z}}] = \text{Num_Int_2}(\mathbf{q}(0), \dot{\mathbf{z}}(0))$

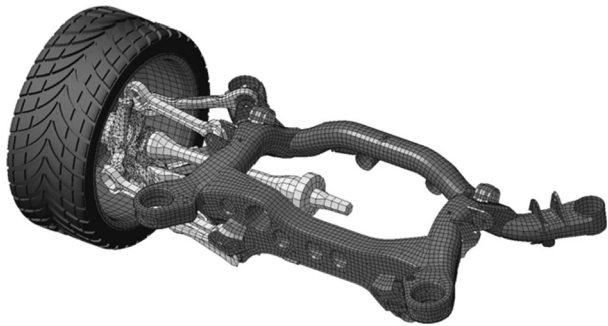
for $i = 1$ to i_{end}

$[\Delta \mathbf{q}_{(i)} \ \Delta \dot{\mathbf{z}}_{(i)}] = \text{Function_Evaluation_2}(\mathbf{q}_{(i)}, \dot{\mathbf{z}}_{(i)}, t_{(i)}, \mathbf{B})$

$$\begin{bmatrix} \Delta \mathbf{q}_{(i)}^* - \Delta \mathbf{q}_{(i)} \\ \boldsymbol{\mu}_{(i)} \end{bmatrix} = \begin{bmatrix} \mathbf{M}_{(i)} & h \boldsymbol{\Phi}_{(i)}^T \\ \boldsymbol{\Phi}_{(i)} & \mathbf{0} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{0} \\ -\boldsymbol{\Phi}(\mathbf{q}_{(i)} + \Delta \mathbf{q}_{(i)}, t_{(i+1)}) \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{q}_{(i+1)} \\ \dot{\mathbf{z}}_{(i+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{q}_{(i)} \\ \dot{\mathbf{z}}_{(i)} \end{bmatrix} + \begin{bmatrix} \Delta \mathbf{q}_{(i)}^* \\ \Delta \dot{\mathbf{z}}_{(i)} \end{bmatrix}$$

Fig. 1 Rear multilink suspension of a rear-wheel-drive passenger car



elements which are the differential shaft and the halfshaft. The differential shaft is connected to the chassis by means of a cylindrical joint and to the halfshaft by means of a universal joint. The halfshaft–rim connection is also modelled as a universal joint. Finally, the rim is connected to the knuckle by means of a revolute joint. The shock-absorber is modelled as a spring-damper force element with linear stiffness and damping characteristics. A nonlinear tire force element has also been introduced, in which the tire–road interaction forces applied to the rim are functions of the indentation between the tire contact-patch and the road surface. Tangential forces at the tire–road contact patch are computed by the Pacejka's magic formulas, while the single contact point transient tire model [14] has been implemented in order to take into account the dynamic phenomena related to the tire carcass compliance.

Starting from the common mechanical system described above, 2 models have been obtained. In the first one, which will be referred to as *bushings model*, the connections of the 5 suspension's links with the subframe and the knuckle are defined using 10 bushing force elements with linear stiffness and damping properties. In the second model, which will be indicated as *ideal joints model*, the suspension's links are connected to the subframe by means of 5 universal joints and to the knuckle by means of 5 spherical joints. The *bushings model* and the *ideal joints model* will be used to assess the efficiency of Algorithms 1 and 2 when dealing with stiff DAEs problems. In particular, their performances will be first

Table 4 Résumé of the *bushings model* and the *ideal joints model* properties

	Bushings model	Ideal joints model
# of configuration parameters	$n = 77$	$n = 77$
# of constraint equations	$m = 31$	$m = 66$
# of DOFs	$f = 46$	$f = 11$

tested in the case of a classical suspension model (*bushings model*) and then in the case of a suspension model with a high number of ideal joints connections (*ideal joints model*).

In order to formulate the EOMs of the two suspension models under analysis, a dependent Cartesian coordinates' approach has been adopted, in which the orientation of each body is described by means of 4 dependent Euler parameters. The total number of configuration parameters required to completely describe the position and orientation of each body is thus $n = 77$. The two models have 31 common constraint equations which are the 11 constraint equations imposing the relationships between the dependent Euler parameters; the 3 constraint equations defined to eliminate the rotational DOFs of the chassis; the 17 constraint equations associated to the ideal joints applied among the driveline elements. In the ideal joints model, 5 additional universal joints are employed to connect the suspension's links to the rear subframe and 5 additional spherical joints are used to connect the links to the knuckle. 35 additional constraint equations are thus needed for the definition of the *ideal joints model* as resumed in Table 4.

The EOMs of the two suspension models have been obtained symbolically through the Lagrange's equations command available in the Maple library proposed by Lot et al. in [15]. Once the EOMs have been derived, all the terms needed to perform the function evaluations in Algorithms 1 and 2 have been isolated and then exported in the Matlab environment. In particular, the following terms have been extracted:

- The mass matrix $\mathbf{M}(\mathbf{q})$;
- The vector of generalized applied forces $\mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}, t)$;
- The vector containing the constraint equations in residual form $\Phi(\mathbf{q})$;
- The Jacobian of the constraint equations $\Phi_{\mathbf{q}}(\mathbf{q})$;
- The vector $\mathbf{c}(\mathbf{q}, \dot{\mathbf{q}})$ coming from the double differentiation of the constraint equations;
- The Jacobians of the vector of generalized applied forces $\mathbf{J}_{\mathbf{q}(t)}(\mathbf{q}, \dot{\mathbf{q}}, t)$ and $\mathbf{J}_{\dot{\mathbf{q}}(t)}(\mathbf{q}, \dot{\mathbf{q}}, t)$.

Note that, since in the system under analysis all the constraint equations are time independent (scleronomous constraints), the term containing the constraint conditions Φ , its Jacobian $\Phi_{\mathbf{q}}$ and the vector \mathbf{c} do not explicitly depend on time. Furthermore, the term Φ_t vanishes. The Jacobians required by the linearly implicit Euler methods are computed symbolically by taking into account the stiff terms in the vector of generalized applied forces which, in the suspension models under analysis, consist only of the reaction forces of the 4 bushings connecting the subframe to the chassis (both *ideal joints model* and *bushings model*) and of the 10 bushings connecting the suspension's links to the subframe and to the knuckle (*bushings model* only). The availability of symbolical expressions for the Jacobians of the vector of generalized applied forces is in line with what is actually implemented in the commercial multibody software LMS VirtualLab.Motion where analytical expressions are available for an efficient evaluation of the exact Jacobians [16].

6.2 Numerical results: computational time and accuracy

In the simulated test manoeuvre, the vehicle settles on a flat road and then, from the rest condition, it accelerates due to a step input torque of 100 Nm applied from the chassis to

Table 5 Computational time of Algorithm 1

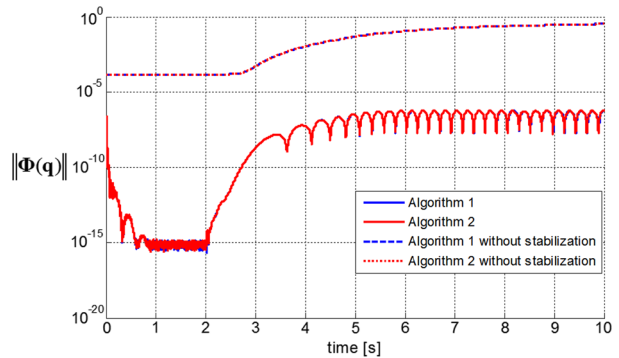
Algorithm 1	Bushings model		Ideal joints model	
	Solving steps	Time [s]	% of time	Time [s]
$\Phi_{\mathbf{q}(i+1)}$	0.24	8.4	0.38	7.5
$[\Delta \dot{\mathbf{q}}(i) \ \lambda_{(i)}]^T$	2.61	91.6	4.68	92.5
Total Time	2.85		5.06	

Table 6 Computational time of Algorithm 2

Algorithm 2	Bushings model		Ideal joints model	
	Solving steps	Time [s]	% of time	Time [s]
$\mathbf{T}_{(i)} = [\mathbf{S}_{(i)} \ \mathbf{R}_{(i)}] = \begin{bmatrix} \Phi_{\mathbf{q}(i)} \\ \mathbf{B} \end{bmatrix}^{-1}$	2.44	68.3	2.47	66
$\dot{\mathbf{q}}_{(i)} = \mathbf{T}_{(i)} \begin{bmatrix} \mathbf{0} \\ \dot{\mathbf{z}}_{(i)} \end{bmatrix}$	0.1	2.8	0.11	2.9
$\mathbf{c}_{(i)}$	0.29	8.1	0.86	23
$\mathbf{S}_{(i)} \mathbf{c}_{(i)} = \mathbf{T}_{(i)} [\mathbf{c}_{(i)} \ \mathbf{0}]^T$	0.07	2	0.08	2.1
$\Delta \dot{\mathbf{z}}_{(i)}$	0.67	18.8	0.22	5.9
Total Time	3.57		3.74	

the differential shaft. After the acceleration phase the vehicle incurs in a square bump with 2 cm height and 2 m length. The total simulated time is of 10 seconds. Algorithms 1 and 2 have been used for the numerical integration of the EOMs associated to the *bushings model* and the *ideal joints model* in the test manoeuvre scenario under analysis.

In order to evaluate the performances of the 2 algorithms, the non-common operations in the corresponding pseudo-codes shown in Sects. 3 and 4 have been isolated and the associated computational times have been checked. Since the code has been implemented in Matlab interpreted language, the absolute CPU-times are only indicative of the performance. The percentage time and relative differences in absolute time can better be used to address the differences between the two algorithms. In Table 5, the solving steps exclusively required by Algorithm 1 are reported. In particular, the computational time required by each operation during the numerical integration of the *bushings model* and the *ideal joints model* is shown. It can be observed that the computational time required to find the variations in the dependent velocities and the Lagrange multipliers at the current time step greatly increases when the ideal joints model is considered. Indeed, when Algorithm 1 is employed, an increase in the number of constraint equations implies the solution of a bigger system of linear equations at each time step. In the case under analysis, the dimension of the linear system to be solved at each time step increases from $n + m = 108$ in the case of the *bushings model* to $n + m = 143$ for the *ideal joints model*. The solving steps which are only related to Algorithm 2 and the corresponding computational times employed for integrating the EOMs of the *bushings model* and of the *ideal joints model* are reported in Table 6. As pointed out before, the computation of the projection matrix \mathbf{R} requires the inversion of an $n \times n$ matrix at each time step. However, the associated computational burden remains constant as the number of constraint equations increases when moving from the *bushings model* to the *ideal joints model*. On the other hand, the dimension of the linear system to be solved at each time step to find the variations in the independent velocities decreases as the number

Fig. 2 Constraint violations

of constraint equations increases. In the case under analysis, it drops from $n - m = 46$ in the *bushings model* to $n - m = 11$ in the *ideal joints model*. The additional numerical burden to evaluate the projection matrix \mathbf{R} in Algorithm 2 is thus compensated by a great reduction in the computational time required to find the variations in the independent velocities when the highly-constrained mechanical system represented by the *ideal joints model* is considered. Indeed, by cross-checking the total computational times in Tables 5 and 6, it can be noticed that the automated switching of the EOMs to an independent coordinates' representation improves the efficiency of the linearly implicit Euler method in the *ideal joints model* case.

Figure 2 shows the norm of the residuals in the constraint equations during the 10 seconds of simulation. For both Algorithms 1 and 2 the errors in the constraint conditions remain bounded throughout the whole simulation if the non-iterative projection step described in Sect. 5 is applied. On the contrary, when the stabilization step is not applied (dotted lines), an unbounded growth of the error in the constraint conditions takes place.

In order to obtain a reference solution, a co-simulation has been set up between the commercial multibody software LMS VirtualLab.Motion and Simulink. The co-simulation set-up allows for the use of the same tire model in both the reference simulation and the test simulations obtained by means of Algorithms 1 and 2. In particular, the nonlinear tire model described previously has been implemented in a Simulink block and integrated in the remaining part of the suspension modelled using LMS VirtualLab.Motion. Two reference solutions have been obtained, one for the *bushings model* and one for the *ideal joints model*. To verify the accuracy of the solutions obtained using Algorithms 1 and 2, the displacements, velocities and accelerations at the CoG of the chassis along the vertical direction are compared with the reference solution in Fig. 3. A zoom of the responses in the time window between 8 and 9.5 s is shown in the graphs on the right, which highlights the vertical response of the chassis when the wheel hits the bump. Results reported in Fig. 3 show that the responses obtained using Algorithms 1 and 2 correctly match the reference solution. Note that in Fig. 3 and in the rest of the paper only the simulation results related to the *bushings model* will be considered since they show the same level of accuracy of results related to the *ideal joints model*.

The bushing reaction forces at the fore-left bushing connection between chassis and sub-frame are compared in Fig. 4. When the wheel hits the bump, the results obtained with both implementations of the linearly implicit method poorly match the reference solution as can be appreciated in the zooms shown in the right graphs. This is in line with what has been shown in the literature on the accuracy of the first order linearly implicit Euler integration method [4]. The same behaviour has been observed for the bushing reaction torques.

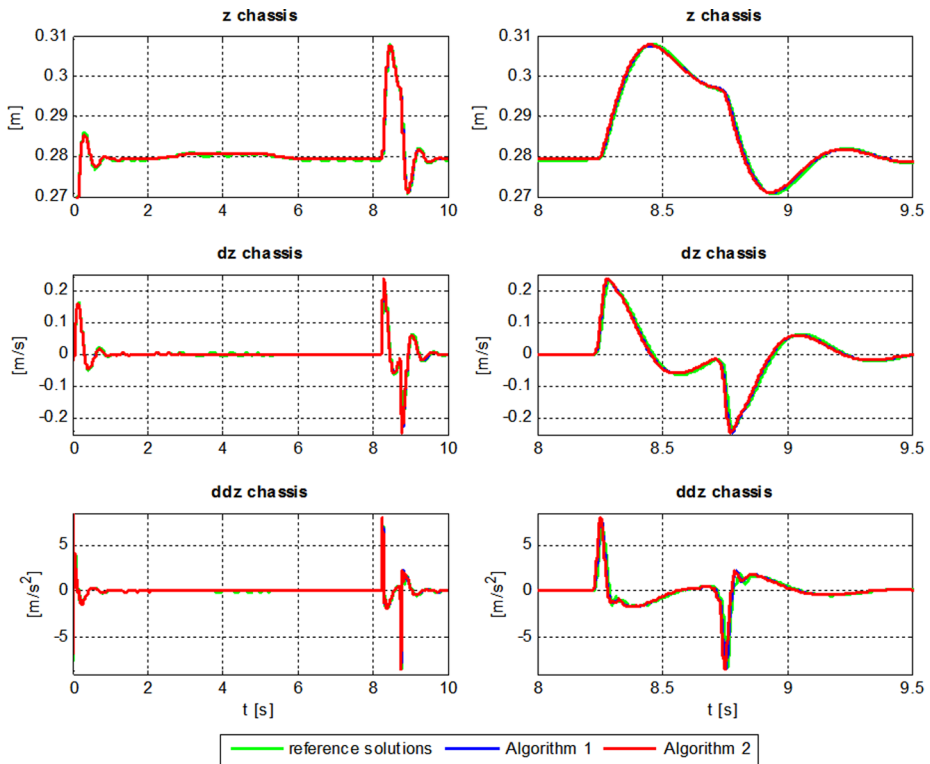


Fig. 3 Vertical position, velocity and acceleration at the chassis CoG

7 Summary

In this paper, we investigate the improvements in the efficiency of the linearly implicit Euler method coming from an independent coordinates' representation of the EOMs associated to mechanical systems containing kinematic closed loops with ideal joints and stiff force elements. The automatic transformation of the EOMs from a dependent to an independent coordinates' formulation at each time step relies on the method based on the matrix \mathbf{R} whose columns represent a basis of the null-space of the constraint Jacobian matrix. Proper stabilization of the constraint equations is carried out by means of a non-iterative projection method which guarantees the drift-off effect to remain bounded for arbitrarily long simulations.

Numerical results show that the proposed implementation of the linearly implicit Euler method may enhance the efficiency of the integration process when dealing with highly constrained multibody systems described by a considerable number of configuration parameters. The accuracy level of the numerical solution has been proved to be equivalent to that achieved with the classical implementation of the linearly implicit Euler method. Therefore, the proposed algorithm seems particularly attractive for RT automotive applications where an automated switching from a dependent to an independent coordinates' representation of the EOMs translates into a considerable reduction in the dimensions of the linear system to be inverted at each time step in order to proceed with the integration process.

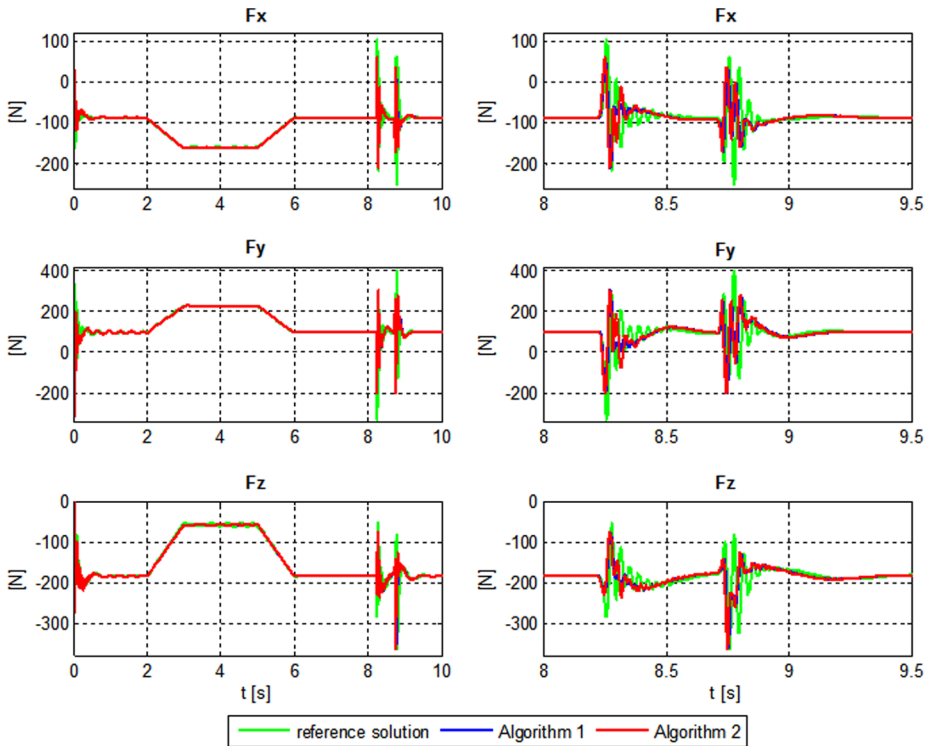


Fig. 4 Reaction forces at the fore-left bushing connection

Acknowledgements We gratefully acknowledge the European Commission (EC) for its support of the Marie Curie IAPP project 285808 “INTERACTIVE” (“Innovative Concept Modelling Techniques for Multi-Attribute Optimization of Active Vehicles”). Furthermore, we kindly acknowledge IWT Vlaanderen for its support of the ongoing research project “Model Driven Physical Systems Operation—MODRIO”, which is part of the ITEA2 project 11004 “MODRIO” (in turn, supported by the European Commission).

References

1. Eichberger, A., Rulka, W.: Process save reduction by macro joint approach: the key to real time and efficient vehicle simulation. *Veh. Syst. Dyn.* **41**, 401–413 (2010)
2. Weber, S., Arnold, M.: Model reduction via quasistatic approximations. *Proc. Appl. Math. Mech.* **10**, 655–656 (2010)
3. Burgermeister, B., Arnold, M., Eichberger, A.: Smooth velocity approximation for constrained systems in real-time simulation. *Multibody Syst. Dyn.* **26**, 1–14 (2011)
4. Arnold, M., Burgermeister, B., Eichberger, A.: Linearly implicit time integration methods in real-time applications: DAEs and stiff ODEs. *Multibody Syst. Dyn.* **17**, 99–107 (2007)
5. Burgermeister, B., Arnold, M., Esterl, B.: DAE time integration for real-time applications in multi-body dynamics. *Z. Angew. Math. Mech.* **86**, 759–771 (2006)
6. Rill, G.: A modified implicit Euler algorithm for solving vehicle dynamic equations. *Multibody Syst. Dyn.* **15**, 1–24 (2006)
7. Schiela, A., Bornemann, F.: Sparsing in real time simulation. *Z. Angew. Math. Mech.* **83**, 637–647 (2003)
8. García de Jalón, J., Bayo, E.: *Kinematic and Dynamic Simulation of Multibody Systems: The Real-Time Challenge*. Springer, New York (1994)

9. Cuadrado, J., Cardenal, J.: Modeling and solution methods for efficient real-time simulation of multibody dynamics. *Multibody Syst. Dyn.* **1**, 259–280 (1997)
10. Baumgarte, J.: Stabilization of constraints and integrals of motion in dynamical systems. *Comput. Methods Appl. Mech. Eng.* **1**, 1–16 (1972)
11. Hairer, E., Wanner, G.: *Solving Ordinary Differential Equations. II. Stiff and Differential-Algebraic Problems*, 2nd edn. Springer, Berlin Heidelberg New York (1996)
12. Pennestrì, E., Valentini, P.P.: Coordinate reduction strategies in multibody dynamics: a review. In: *Conference on Multibody System Dynamics*, Pitesti, Romania, 25–26 October (2007). ISSN 1582-9561
13. Lubich, Ch., Engstler, Ch., Nowak, U., Pöhle, U.: Numerical integration of constrained mechanical systems using MEXX. *Mech. Struct. Mach.* **23**, 473–495 (1995)
14. Pacejka, H.B.: *Tire and Vehicle Dynamics*, 2nd edn. Elsevier, Oxford (2006)
15. Lot, R., Da Lio, M.: A symbolic approach for automatic generation of the equations of motion of multibody systems. *Multibody Syst. Dyn.* **12**, 147–172 (2004)
16. Prescott, B., Heirman, G., Furman, M., De Cuyper, J.: Using High-Fidelity Multibody Vehicle Models in Real-Time Simulations. SAE Technical Paper 2012-01-0927 (2012)