Journal of Mechanical Science and Technology, Vol. 19, No. 1, ⟨Special Edition⟩ pp. 461~472, 2005 **461**

A Modular Formulation for Flexible Multibody Systems Including Nonlinear Finite Elements

Lars Kübler*
Institute of Applied Mechanics, University of Erlangen-Nuremberg, Egerlandstr, 5, 91058 Erlangen, Germany
Peter Eberhard
Institute B of Mechanics, University of Stuttgart, Pfaffenwaldring, 9, 70569 Stuttgart, Germany

A formulation for flexible multibody systems (MBS) is investigated, where rigid MBS substructures are coupled with flexible bodies described by a nonlinear finite element (FE) approach. Several aspects that turned out to be crucial for the presented approach are discussed. The system describing equations are given in differential algebraic form (DAE), where many sophisticated solvers exist. In this paper the performance of several solvers is investigated regarding their suitability for the application to the usually highly stiff DAE. The substructures are connected with each other by nonlinear algebraic constraint equations. Further, partial derivatives of the constraints are required, which often leads to extensive algebraic transformations. Handcoding of analytically determined derivatives is compared to an approach utilizing algorithmic differentiation.

Key Words: Flexible Multibody System, Large Deformation, MBS, FEM, Algorithmic Differentiation, Numerical Simulation of DAE, DAE Solver

1. Introduction

A formulation for flexible MBS is discussed where rigid MBS substructures are coupled with flexible bodies described by a nonlinear FE approach. For the FE description absolute coordinates are used for isoparametric hexagonal elements under consideration of structural damping for hyper-elastic materials of neo-Hookean type. Hexagonal elements are for example important if general solid bodies with low stiffness, i.e. not necessarily negligible large deformations, are part of the MBS and cannot be modelled using beam, plate, or shell elements. The use of absolute coordinates together with a Lagrangian FE formulation allows for large deformations and provides an accurate description of rigid body motion and inertia in the case of large rotations. Further, constant mass matrices are obtained for isoparametric elements.

The governing equations of the system are given in descriptor form as a set of DAE which is usually of substantial stiffness and has to be solved appropriately. Several sophisticated solvers exist and have been subject to many investigations concerning their performance for rigid MBS or flexible MBS with small deformations, typically using floating frame of reference formulations. Here, an appropriate choice of existing DAE solvers for the application to our flexible MBS is discussed, where especially the performance with respect to accuracy of the solution and numerical expense is of interest.

The substructures building the global system

* Corresponding Author,
E-mail : kuebler@ltm.uni-erlangen.de
TEL : +49-9131-8528508; **FAX** : +49-9131-8528503
Institute of Applied Mechanics, University of Erlangen-Nuremberg, Egerlandstr. 5, 91058 Erlangen, Germany.
(Manuscript **Received** November 29, 2004; **Revised** December 15, 2004)

are connected to each other using a constraint formulation which leads to another important aspect. The nonlinear algebraic constraint equations and their necessary partial derivatives are provided analytically which often requires extensive algebraic transformations, especially for connections to flexible bodies, and is due to a high complexity strongly error prone. Therefore, in this paper also algorithmic differentiation is utilized for reliable computation of the derivatives.

2. Modular Organization of the Flexible Multibody System

In our approach the global flexible MBS is organized in modules, as illustrated in Figure 1. Several subsystems that can be rigid bodies, flexible bodies or complete loop-free rigid MBS substructures are assembled to the global model. The description of flexible bodies is based on Bathe (1995) and Wriggers (2001). Details on our specific description and further on the implemented strain dependent material damping are given in Kubler, Eberhard and Geisler (2003).

The motion of the subsystems is kinematically constrained by mechanical joints or kinematic drivers, both described by nonlinear algebraic constraint equations. While the relative motion of the subsystems is described in Cartesian space, it is possible to define rigid MBS substructures using relative coordinates.

This approach utilizes on the one hand the efficiency of a formulation with relative coordinates, see Schiehlen and Eberhard (2004). On the other hand the approach allows for a high level of generality by assembling the subsystems under application of a constraint formulation with the advantage that the formulation of the equations of motion even for complex systems is straightforward. This approach is open to the addition of various complex system components, including MBS with kinematic loops. For example, this feature proofs to be very advantageous when flexible bodies are interconnected with other subsystems, where it is only necessary to provide the finite element description of a free body in absolute coordinates. Complex interactions like inertia coupling between rigid substructures and flexible bodies are considered implicitly via the constraint equations.

However, since in this approach flexible bodies are described with isoparametric hexagonal elements, only nodal translational degrees of freedom are available for the description of relative orientations of flexible bodies with respect to rigid or other flexible bodies. Therefore, a reference frame at connection points has to be defined appropriately, as discussed in Kubler, Eberhard and Geisler (2003,2).

3. Global Equations of Motion

The global system is assembled from n_s subsystems as illustrated in Figure 1. The subsystems are interconnected by n_j joints with their constraint equations summarized in the global constraint vector $\boldsymbol{c} \in R^{n_c}$.

The equations of motion of the global system can, for example, be derived by application of d'Alembert's principle, which states that the virtual work of reaction forces ${}_d W_R$ is zero for the entire system (including the reaction forces between the subsystems), see e.g. Schiehlen and Eberhard (2004). This can be expressed equivalently with the virtual work of inertia forces

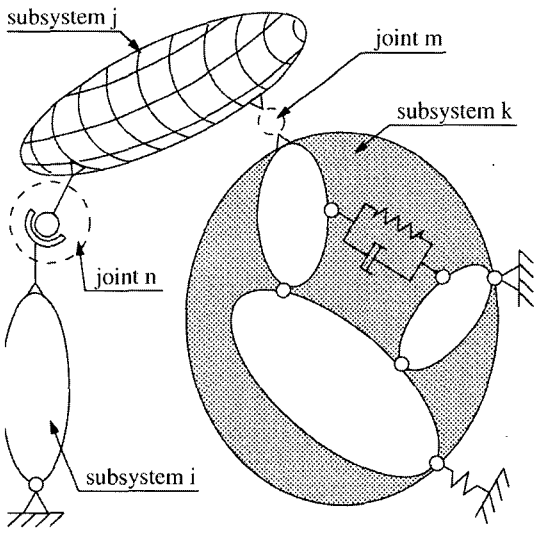

Fig. 1 Modular organization of the global system

and the virtual work of internal and external applied forces ${}_{\mathrm{d}}W_I$ and ${}_{\mathrm{d}}W_A$, respectively

$$\sum_{i=1}^{n_s}\delta W_R^i=0 \Longleftrightarrow \sum_{i=1}^{n}(\delta W_I^i+\delta W_A^i)=0 \qquad (1)$$

Equation (1) can be split up in n_r rigid MBS subsystems and n_f flexible subsystems, i.e. single flexible bodies. For rigid MBS substructures the expression $\delta W_I^i+\delta W_A^i$ can be described in Lagrange's formulation of d'Alembert's principle, compare Eberhard (2000), while for flexible subsystems it can be derived from Cauchy's equation of motion for continua and the principle of virtual work after FE discretization, yielding the nonlinear weak form (here in total Lagrangian description), see Wriggers (2001). It follows in index notation (lower index)

$$\begin{aligned}&\sum_{i=1}^{n_r}\delta y_j^i({}_{\mathrm{r}}M_{jk}^i\ddot{y}_k^i+{}_{\mathrm{r}}k_j^i-{}_{\mathrm{r}}g_j^i)\\&+\sum_{i=1}^{n_f}\delta U_j^i\Big({}_{\mathrm{f}}M_{jk}^i\ddot{U}_k^i-{}_{\mathrm{f}}g_j^i+\int_{\Omega^0}B_{hjk}F_{hl}S_{lk}dV\Big)=0\end{aligned} \qquad (2)$$

where for the rigid subsystems $\mathbf{y}$ denotes the local generalized coordinates vector, ${}_{\mathrm{r}}\mathbf{M}$ is the mass matrix, ${}_{\mathrm{r}}\mathbf{k}$ is the vector of generalized centrifugal and Coriolis forces and ${}_{\mathrm{r}}\mathbf{g}$ is the vector of generalized applied forces. For flexible subsystems there is the nodal displacement vector $\mathbf{U}$, the constant mass matrix ${}_{\mathrm{f}}\mathbf{M}$, the vector of applied forces ${}_{\mathrm{f}}\mathbf{g}$, the constant B-operator, the deformation gradient $\mathbf{F}$ and the 2nd Piola-Kirchhoff stress tensor $\mathbf{S}$. All quantities are explained in detail in Kubler, Eberhard and Geisler (2003).

If we introduce the elastic stiffness vector

$${}_{\mathrm{f}}k_j^i=\int_{\Omega^0}B_{hjk}F_{hl}S_{lk}dV \qquad (3)$$

in (2) it can be seen that both the parts of the equations for rigid and flexible bodies are of the same structure. Therefore, after introduction of according global quantities, e.g. the global coordinates vector

$$\mathbf{q}=\lfloor\,\mathbf{y}^1\;\mathbf{y}^2\;\cdots\;\mathbf{y}^{n_r}\;\mathrm{U}^1\;\mathrm{U}^2\;\mathrm{U}^{n_f}\rfloor \qquad (4)$$

and similarly a global mass matrix $\mathbf{M}$ (block diagonal) and global vectors $\mathbf{k}$ and $\mathbf{g}$ that assemble the corresponding local vectors, it follows from (2) a variational problem with constraints)

$$\delta\mathbf{q}\cdot(\mathbf{M}\cdot\ddot{\mathbf{q}}+\mathbf{k}-\mathbf{g})=0\;\;\forall\,\delta\mathbf{q}:\mathbf{C}_q^T\cdot\delta\mathbf{q}=\mathbf{0} \qquad (5)$$

with the global constraint Jacobian matrix

$$\mathbf{C}_q=\frac{\partial\mathbf{c}}{\partial\mathbf{q}} \qquad (6)$$

The variational problem is solved by the Lagrange multiplier method yielding the equations of motion of the global flexible MBS

$$\begin{aligned}&\mathbf{M}(\mathbf{q},\,t)\cdot\ddot{\mathbf{q}}+\mathbf{k}(\dot{\mathbf{q}},\,\mathbf{q},\,t)-\mathbf{C}_q^T(\mathbf{q},\,t)\cdot\boldsymbol{\lambda}\\&=\mathbf{g}(\dot{\mathbf{q}},\,\mathbf{q},\,t)\end{aligned} \qquad (7)$$

with the not yet determined Lagrange multipliers $\lambda\in R^{n_c}$.

Note that equations (7) are highly nonlinear and due to the coupled model with rigid and elastic components often highly stiff, i.e. the solution is composed of motions on different time scales with low and fast (highly oscillatory) modes.

Equations (7) are to be solved together with the constraint vector $\mathbf{c}=\mathbf{0}$, forming a DAE with differentiation index 3. It is analytically equivalent to use the first time derivative of $\mathbf{c}$

$$\dot{\mathbf{c}}=\mathbf{C}_q\cdot\dot{\mathbf{q}}+\frac{\partial\mathbf{c}}{\partial t}=\mathbf{0} \qquad (8)$$

which yields differentiation index 2, or the second time derivative (index 1)

$$\ddot{\mathbf{c}}=\mathbf{C}_q\cdot\ddot{\mathbf{q}}+\dot{\mathbf{q}}\cdot\frac{\partial\mathbf{C}_q}{\partial\mathbf{q}}\cdot\dot{\mathbf{q}}+2\frac{\partial\mathbf{C}_q}{\partial t}\cdot\dot{\mathbf{q}}+\frac{\partial^2\mathbf{c}}{\partial t^2}=\mathbf{0} \qquad (9)$$

There are various different index definitions, e.g. discussed in Hairer and Wanner (1996), Arnold (1998), Eich-Soellner and Fuhrer (1998), etc. For mechanical systems, mostly the differentiation index, which is defined as the number of differentiations required to obtain an ODE in all unknowns, and the perturbation index, which describes the sensitivities of the equations under perturbations, are used. For the system formulation applied here, both indices are identical. Therefore, in the following we will only speak of the 'index' when distinguishing between the different formulations.

Numerically the different index formulations have strong influence on convergence and effort for solving the nonlinear system, see e.g. Eich-

Soellner and Fuhrer (1998) or Simeon (1994). DAE with index>1, especially index 3 formulations, suffer from worse approximation properties than index 1 DAE, in particular error estimation and step size control become very complex. From that point of view it would be obvious to use an index 1 formulation. However, the algebraic and therefore also the numerical effort to provide the constraint equations increases strongly with decreasing index. It is especially demanding to provide all partial derivatives required for equation (9). Further, in index 1 and index 2 formulations, the position constraint is no longer considered explicitly. Numerical errors like truncation errors lead to a growing violation of the position constraints with increasing time. This so-called drift-off effect is relatively moderate for index 2 formulations (linear growth) but strong for index 1 formulations (quadratic growth), compare Simeon (1994). For both cases stabilization techniques are available to handle the drift instability. As a reasonable trade-off here an index 2 formulation is chosen with moderate algebraic effort for the constraint description, relatively low drift-off and still good approximation properties of the numerical solution. In fact, regarding robustness and efficiency in many examples the index 2 formulation shows very good performance, see e.g. examples in Hairer and Wanner (1996), Arnold (1998), Eich-Soellner and Fuhrer (1998).

4. Constraint Formulation for the Connection of Substructures

Most of the practically used kinematic constraints can be built by setting algebraic relations between vectors defined on the bodies, as discussed in detail in Shabana (1998) or Nikravesh (1988).

For the index 2 formulation, applied here, the partial derivatives $\mathbf{C}_q = \partial \mathbf{c}/\partial \mathbf{q}$ and the time derivatives $\partial \mathbf{c}/\partial t$ are required. These derivatives can be determined analytically, Section 4.1. While this approach allows for fast computation of the derivatives, it is often of substantial complexity to deduce the analytic expressions from the constraint vector $\mathbf{c}$, which, hence, requires large human effort and is error prone. Therefore, as a second approach, also algorithmic differentiation is utilized here, Section 4.2. Algorithmic differentiation offers many benefits, especially if compared to approximational methods like finite differences. It is accurate up to machine precision, applicable even to large, complex computer programs, and can be produced with minimal human effort.

4.1 Handcoded analytic derivatives

Kinematic constraints are built by algebraic relations between vectors of marker frames defined on the bodies, as indicated in Figure 2, where the connection of a rigid body (part of subsystems A) and a flexible body (subsystem B) by a revolute joint at the marker frames B_1 and B_2 is sketched.

In order to determine the quantities $\mathbf{C}_q$ and $\partial \mathbf{c}/\partial t$ analytically, it proved as particularly attractive to transform the orientation matrices $\mathbf{S}_{B1}$ and $\mathbf{S}_{B2}$ at the connection points to unit quaternions, which allow for proper derivation of mathematical relations for the observed coordinate frames. Many basic identities and analytical relations exist, e.g. given in Nikravesh (1988). In Kubler and Eberhard (2002) the required analytical derivatives for constraint vectors were already derived for rigid MBS by extending the approach in Serban and Haug (1988). Further, in Kubler, Eberhard and Geisler (2003) and (2003,2) this approach was augmented to flexible multibody dynamics, i.e. to the connection of

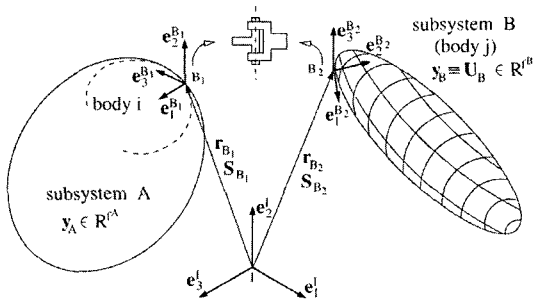

Fig. 2 Schematic representation of the connection of subsystems

flexible bodies described by a nonlinear FE approach.

Generally, it must be considered that determining the analytic derivatives is of high complexity and comes along with large human effort, but once derived and implemented works fine and relatively fast.

4.2 Derivatives by algorithmic differentiation

Algorithmic or automatic differentiation (AD) is introduced, see Bischof, Roh and Mauer (1997) and Griewank (2000), as a powerful technique to compute derivatives of functions given in the form of a computer program in Fortran, C, or C++. "In contrast to traditional approaches such as handcoding of analytic expressions, numerical approximation by divided differences, or manipulation of symbolic algebraic expressions by computer algebra systems, automatic differentiation offers the following substantial benefits: it is accurate up to machine precision, efficient in terms of computational cost, applicable to a 1-line formula as well as to a 100,000-line code, and can be produced with minimal human effort" (Bischof and Bucker (2000)).

AD techniques augment the program with derivative computation by consecutive application of the chain rule of differentiation to elementary operations.

In Section 4.1 the high complexity and human effort of handcoding of the required derivatives was indicated. Above that, this approach is relatively error prone. So, our idea was to use the AD tool ADIC, Bischof, Roh and Mauer (1997), to derive the required quantities with comparable accuracy but with much less human effort and high reliability, see Figure 3.

ADIC employs a source-to-source program transformation approach. The new, automatically generated source code provides exact gradients up to round-off errors and can be compiled and linked to the executable program. In fact, we verified the received derivatives from AD with previously handcoded and checked analytic derivatives for different algebraic relations and found very good agreement. The use of AD allows us to easily define further vector relations for constraints. However, just to give the reader an idea of possible computation times, in our implementation the computation times for the derivatives from the AD approach turned out to be by a factor of approximately four higher than those from handcoding, which also considerably effects the overall computation times (by a factor of about two). Therefore, we went back to the analytically determined derivatives and use the AD approach as a proper, reliable way to verify the handcoded programs.

5. Numerical Solution of DAE

The numerical solution of DAE has been (and still is) subject to intensive research and today many sophisticated solvers are available, which are often based on methods for the numerical solution of ordinary differential equations (ODE) In our program system it can be chosen between three advanced solvers with different underlying methods: R_{ADAU5} (single-step method), D_{DASSL} (multi-step method) and M_{EXAX} (extrapolation method), see Section 5.2.

For the systems observed here, it has to be taken into account that the resulting DAE is due to the coupled model with rigid and elastic components often highly stiff. In particular the description of flexible bodies with a pure FE approach and many degrees of freedom causes solutions with highly oscillatory modes. Many definitions of stiffness exist. For example in Hairer and Wanner (1996) among others a pragmatic declaration is given: "stiff equations are problems for which explicit methods don't work". Another, very perspicuous statement is given in Eich-Soellner and Fuhrer (1998): "It might happen, that for a given problem the step size is limited additionally by a stability bound [···]. If this restricts the step size more than the tolerance bound, the problem is called stiff".

Generally, stiff integrators are A-stable or at least A(a)-stable, implicit and require a corrector iteration based e.g. on Newton's method. The authors want to point out that the solver M_{EXAX} per se has not been developed for stiff systems.

However, since we had very promising results for moderately stiff rigid MBS we wanted to examine its behavior under application to our substantially stiff flexible MBS, too.

The numerical solution of the index 2 formulation, which is used here, yields relatively low drift-off, as discussed in Section 3. Appropriate stabilization techniques are considered in our program system, see Section 5.1.

5.1 Stabilization techniques

Drift-off appears due to numerical errors like truncation errors, as the position constraint is not considered explicitly in index 1 and index 2 formulations. Different approaches exist to overcome this problem, see e.g. Hairer and Wanner (1996) or Arnold (1998). Here, two techniques are used: Gear-Gupta-Leimkuhler (GGL) stabilization and coordinate projection.

5.1.1 GGL stabilization

Gear, Leimkuhler & Gupta (1985) proposed a technique where an extended system is solved. The index 2 formulation is augmented by the position constraints, yielding an over-determined DAE. To obtain a well posed DAE additional Lagrange multipliers $\boldsymbol{\mu}$ can be added. The system in descriptor form follows as

$$\begin{aligned} \dot{\mathbf{q}} &= \mathbf{v} - \mathbf{C}_q^T \cdot \boldsymbol{\mu} \\ \mathbf{M} \cdot \dot{\mathbf{v}} &= \mathbf{g} - \mathbf{k} + \mathbf{C}_q^T \cdot \boldsymbol{\lambda} \\ \mathbf{0} &= \mathbf{C}_q \cdot \dot{\mathbf{q}} + \frac{\partial \mathbf{c}}{\partial t} \\ \mathbf{0} &= \mathbf{c} \end{aligned} \tag{10}$$

By this approach an index 2 formulation is realized where also the position constraints are fulfilled during numerical solution. The system (10) is analytically equivalent to the index 2 formulation, i.e. for the analytic solution it holds $\boldsymbol{\mu}=\mathbf{0}$, while numerically we have $\|\boldsymbol{\mu}\|=O(\mathrm{tol})$.

5.1.2 Coordinate projection

Each integration step consists of the computation of predictor values $\mathbf{q}^0$ solving the index 2 formulation (or index 1 in the general case) which are subjected to drift, i.e. in general the position constraints (and velocity constraints for index 1) are not met. Then, in a corrector step the coordinates $\mathbf{q}^0$ are projected back on the constraint manifold. The projection step must be solved iteratively, for example applying Newton's method. Different projection methods can be distinguished by the direction of the projection, e.g. orthogonal to the constraint manifold.

In our program system a two-step projection approach is implemented, which is taken from Hairer and Wanner (1996) and was originally proposed by Lubich (1991). In the first step the projection on the position constraints is carried out by (simplified) Newton iterations

$$\begin{aligned} \mathbf{M}(\mathbf{q}_n^0) \cdot (\mathbf{q}_n - \mathbf{q}_n^0) + \mathbf{C}_q^T(\mathbf{q}_n^0) \cdot \eta &= \mathbf{0} \\ \mathbf{c}(\mathbf{q}_n) &= \mathbf{0} \end{aligned} \tag{11}$$

where $\mathbf{q}_n$ results as solution of the nonlinear system. It follows a projection on the velocity constraint by solution of the linear system

$$\begin{aligned} \mathbf{M}(\mathbf{q}_n) \cdot (\dot{\mathbf{q}}_n - \dot{\mathbf{q}}_n^0) + \mathbf{C}_q^T(\mathbf{q}_n) \cdot \eta &= \mathbf{0} \\ \dot{\mathbf{c}}(\mathbf{q}_n) &= \mathbf{0} \end{aligned} \tag{12}$$

Despite of the velocity constraints being included, for index 2 formulations also (12) has to be solved as the velocities after position projection need not to be on the velocity constraint manifold. However, the computational effort can be reduced by evaluating (12) only after several time steps.

In principle, this projection method can be applied to any discretization method. However, for BDF-formulae as used in $\mathrm{D}_{\mathrm{DASSL}}$ it is rather unfavorable as this may act like 'impacts' on the interpolation polynomial. Here, the application of a GGL-stabilization is advantageous.

5.2 DAE-Solvers

In our program system it can be chosen between three advanced DAE solvers: $\mathrm{R}_{\mathrm{ADAU}}5$ (single-step method), $\mathrm{D}_{\mathrm{DASSL}}$ (multi-step method) and $\mathrm{M}_{\mathrm{EXAX}}$ (extrapolation method), which will be briefly introduced in this section.

5.2.1 $\mathrm{R}_{\mathrm{ADAU}}5$

The code $\mathrm{R}_{\mathrm{ADAU}}5$ is based on an implicit Runge-Kutta method (Radau IIa) of 5^{th} order.

This method is L-stable and, therefore, well suited for stiff DAE. A detailed description of the method is given in Hairer and Wanner (1996) and Hairer, Lubich and Roche (1989), where also its convergence behavior under application to DAE of different index is described. Additionally in Hairer and Wanner (1999) Radau methods of different orders are discussed.

$R_{ADAU}5$ is capable of solving index 1, 2 and 3 problems. As stabilization is not included, our own implementations of the GGL-stabilization and coordinate projection are added and used, see Section 5.1.

5.2.2 D_{DASSL}

The solver D_{DASSL}, described in detail in Brenan, Campbell and Petzold (1989), applies backward differentiation formulae (BDF) for the solution of the DAE given in implicit form.

BDF formulae are an important class of multistep methods, based on interpolating the solution points instead of the derivatives. Compared with Adams formulae, they have the advantage that they are stable at infinity and, hence, can also be applied to (semi-explicit) index 2 DAE, see Eich-Soellner and Fuhrer (1998). Further, BDF methods allow to vary the order of the method in a simple way. They are defined in terms of interpolation polynomials based on certain numbers of points. Changing the number of interpolation points changes the order of the polynomial and, therefore, the order of the method.

D_{DASSL} uses an automatic order variation approach with BDF formulae of order one through five, which is also convenient for the begin of the computation, where it can be started with a single step method, then switched to the two step method and so on. After every successful step the most appropriate order for the next step is chosen. It is further possible for the user to define a maximal order of the method. This can be useful for stiff DAE, where it may happen that the step size is stronger restricted by the stability bound of the method than by the tolerance bound. BDF-1 and -2 are A-stable methods and well suited for stiff problems. For higher order BDF formulae A(a)-stability is introduced, which states that the methods are stable for all eigenvalues x of a linear test equation with Re $x<0$ and $|\arg(-x)|<a$, see e.g. Eich-Soellner and Fuhrer (1998). The angles a are given in Table 1. For BDF-3 and higher the above mentioned effect is possible for stiff systems, especially with low or no damping, see Section 6.

Table 1 A(a)-Stability of BDF methods

BDF	1	2	3	4	5
a [°]	90	90	86	73	51

D_{DASSL} was designed for problems of index 1 or lower. However, it can also solve our index 2 formulation by adjusting the error control for the algebraic variables l which are only approximated by order k-1 for BDF-k formulae (they are simply removed from the error control). Stabilization is not included, therefore again our own implementation of the GGL-stabilization is used. As discussed in Section 5.1 coordinate projection is not applied here.

5.2.3 M_{EXAX}

The solver M_{EXAX} (the former M_{EXX}) is an extrapolation method with an underlying half-explicit time stepping (mid-point rule), which is explicit in the differential equations and linearly implicit in the nonlinear constraints, see Lubich et al. (1995). Position and velocity constraints are enforced during the integration by an internal coordinate projection approach. The maximum number of extrapolation steps in each time step is twelve and can be limited by the user to lower numbers.

6. Example

The three-dimensional motion of a five body mechanism consisting of one flexible body and four rigid bodies connected by different types of joints is analyzed, see Figure 4. The flexible body consists of a very soft sample material with the following properties: Young's modulus $E=7\cdot10^5$ N/m2, Poisson ratio n=0.3 and density r=920 kg/m^3. For the elements at the upper and lower tips, where the joints are connected, a stiffer

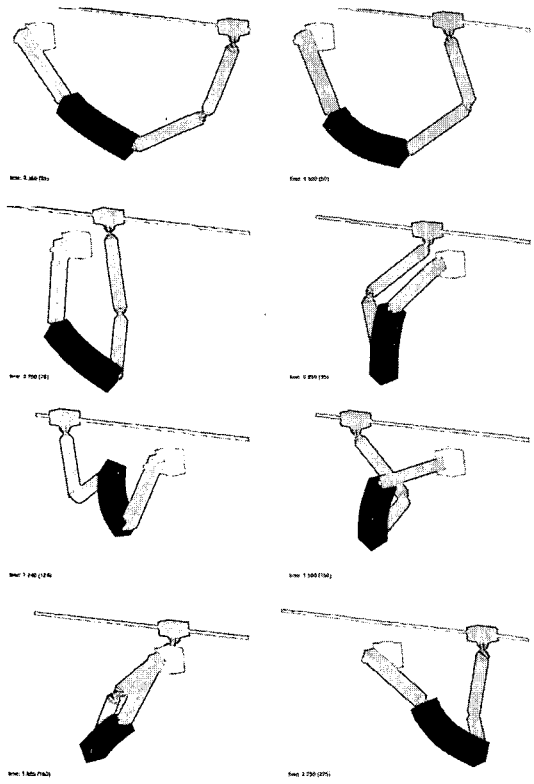

Fig. 4 Frames from an animation of the five body mechanism

material is chosen with $E=10^7\ N/m^2$, n=0.3 and $r=1000\ kg/m^3$. Both materials can be exposed to material damping with the dynamic viscosity $h=1000\ kg/(ms)$.

This example was used to verify the joint descriptions and it turned out that they are fulfilled in a sound way. Further, it was utilized in order to verify the performance of the different DAE solvers, introduced in Section 5, under application to a stiff flexible MBS. Some interesting aspects of our results will be discussed in the following.

In Figure 5 exemplary the position in x-direction of the middle node of the flexible body is given over time. The results for all solvers are here in very good agreement within the given tolerances. Therefore, it is only one curve shown for both cases: with and without material damping.

Even results with and without stabilization

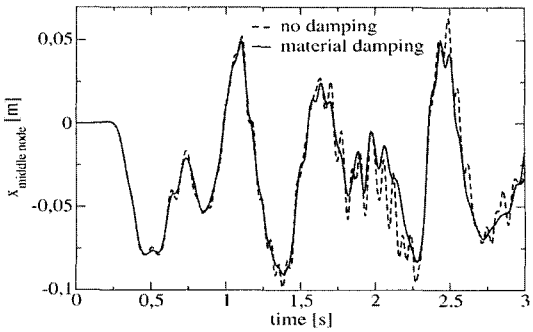

Fig. 5 X-Position of the middle node of the flexible body with and without material damping

correspond very well for the index 2 DAE. The maximal deflection due to drift was in this case below 10^{-6} m. If the curves with and without damping are compared, substantial differences in the solution can be seen. This indicates that for such materials of very low stiffness usually material damping cannot be neglected.

Omitting the damping also strongly effects the computational cost. In the damped case excited high frequency modes are damped out fast, in agreement to the real, physical material behavior. In the undamped case the high frequencies remain in the body motion with two possible effects. Firstly, if these modes have not negligible large amplitudes, i.e. larger than the demanded tolerance, they force the step size to decrease for all methods, hence leading to unphysical results and higher computation times. Secondly, if the amplitudes are negligibly small, A-stable methods will be able to choose the step sizes according to the demanded tolerance and, hence, the solution need not to include these high frequency oscillations. This behaviour can be interpreted as a kind of slight 'tolerance induced' damping. However, not A-stable methods might not be able to choose step sizes according to the tolerance bound, as for stiff systems they might be limited by the stability bound which may force the step size to decrease to very low values yielding high computation times.

n this context the possibility to define a maximal order for the solver D_{DASSL} was observed, considering the stability regions of the different BDF formulae. as discussed in Section 5.2.2.

BDF formulae with an order greater than two are A(a)-stable with a lower than 90° Therefore, for stiff systems the stability bound may lead to lower step sizes than required by the tolerance bound. In Table 2 some statistics from a computation of the five-body mechanism with material damping for limited BDF orders $p_{max}=2$, 3 and 5 and rtol=atol=10^{-8} are given. The reached tolerance, or rather the error (ERR), is estimated by comparison to a reference solution computed for high tolerances with different methods. According to the error estimation of the solvers, see Simeon (1994), ERR follows as

$$\mathrm{DRR}(x):=\sqrt{\frac{1}{n}\sum_{j=1}^{n}\left(\frac{x_j-x_j^{ref}}{rtol\cdot|x_j^{ref}|+atol}\right)^2}\cdot rtol \quad (13)$$

The Jacobian matrix evaluations appear from the Newton iteration and are often very expensive. Normally, the Jacobian matrices change only slowly in time. Therefore, they are usually computed only after several steps in order to save computation time. In Table 2 the ratio of rejected steps to computed steps and the number of Jacobian matrix evaluations, which both increase substantially with increasing maximal order, indicate that for higher orders than two the stability bound influences the step sizes. Further, step size control and automatic order control are competing, so that e.g. in the case $p_{max}=5$ the algorithm changes frequently between different orders, trying to enlarge the stability region by reducing the order and then trying to increase the step width by increasing the order. This also affects the computation times, which are increasing with increasing maximal order.

Table 2 Computation of the example with material damping using ddassl with limited BDF-orders, rtol=atol=10^{-8}

D_{DASSL}	$p_{max}=2$	$p_{max}=3$	$p_{max}=5$
ERR	$4\cdot10^{-7}$	$5\cdot10^{-8}$	$4\cdot10^{-8}$
computed steps (NST)	23672	9434	12557
rejected steps/NST·100%	9.9%	21.3%	38.6%
Jacobian matrix evaluations	233	335	419
normalized CPU-time	1	1.25	1.54

Our conclusions here are, that for highly stiff systems it is inevitable to use the maximal order limitation. While in the case of a relatively high structural damping $p_{max}=3$ can also be interesting, for low or even no damping at all $p_{max}=2$ is the best choice, which is therefore used in all following computations. The effect of increasing computation times becomes even stronger when systems without damping are observed. For example we found for above example without damping that the computation time for $p_{max}=5$ was by a factor of 4.3 higher than that with $p_{max}=2$ with approximately the same accuracy.

In Table 3 some computation statistics are compared for the five-body mechanism with material damping for different solvers, stabilization methods and tolerances. Computation times are normalized with respect to the lowest time received with D_{DASSL} without stabilization for rtol=atol=10^{-6}. The authors want to point out that the results given here are those of one specific example with certain solver parameters (adjusted by the authors to the best of their knowledge) and, therefore, the results might not be representative for all systems or situations. Furthermore, our intention is not to select one solver, for example the fastest one, but to discuss some interesting aspects and gain knowledge and experience.

If the results for the solvers D_{DASSL} and R_{ADAU5} are regarded, it can be seen that the computation times without applied stabilization methods are slightly higher than that with GGL-stabilization, where a larger system has to be solved. However, as a stronger increase was expected, stabilization seems to reduce the computational difficulty, which is for R_{ADAU5} also indicated by the lower numbers of Jacobian matrix evaluations with GGL.

Another observation, confirming the statement in Section 3 that the index 2 formulation shows relatively little drift, is that also the non-stabilized methods mostly reach the high demanded tolerances. However, it should be mentioned here, that we did not expect the drift to be that low.

Table 3 Statistics for the computation with material damping using different solvers (TOL=rtol=atol, NST : number of computed steps, NAC : number of accepted steps, NRJ : number of rejected steps, NFE : number of function evaluations, NJE : number of Jacobian matrix evaluations, CPU : normalized CPU-time)

R_{ADAU5}	TOL=10^{-6}		TOL=10^{-8}	
	no stabil.	GGL	no stabil.	GGL
ERR	$3\cdot10^{-7}$	$3\cdot10^{-8}$	$1\cdot10^{-9}$	$1\cdot10^{-9}$
NST	456	487	876	876
NAC	367	426	855	851
NFE	9736	10178	16158	15118
NJE	310	287	294	292
CPU	2.3	2.4	2.5	2.6

D_{DASSL} (p_{max}=2)	TOL=10^{-6}		TOL=10^{-8}	
	no stabil.	GGL	no stabil.	GGL
ERR	$1\cdot10^{-5}$	$7\cdot10^{-6}$	$4\cdot10^{-7}$	$4\cdot10^{-7}$
NST	5714	5821	23616	24168
NAC	4747	4803	22015	21144
NJE	185	189	222	242
CPU	1	1.0	1.3	1.5

M_{EXAX}	TOL=10^{-6}	TOL=10^{-8}
	coordinate projection	
ERR	$3\cdot10^{-9}$	$3\cdot10^{-11}$
NST	7164	4839
NAC	5269	4421
CPU	2.9	5.0

For other examples larger (but still moderate) drift-off was observed and stabilization was required to obtain high tolerances.

The solver M_{EXAX} reaches results of high accuracy but requires higher computation times, even if results with the same reached tolerances are compared. As already discussed in Section 5, the solver M_{EXAX} with an underlying half-explicit method per se has not been developed for stiff systems and was only applied to the stiff flexible MBS as we had very promising results for moderately stiff rigid MBS. So, as expected for highly stiff systems, D_{DASSL} and R_{ADAU5} should be preferred.

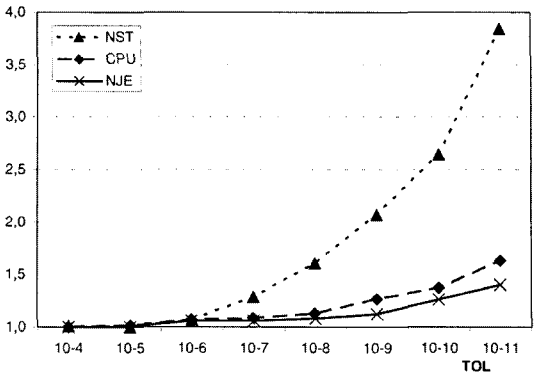

Fig. 6 Computation of the example with material damping using R_{ADAU5} with GGL for different tolerances (shown are normalized values for NST, NJE and CPU)

In principle, the results received from all solvers show the expected behaviour of increasing CPU-times with higher tolerances. For example in Figure 6 results for R_{ADAU5} with GGL stabilization for tolerances TOL from 10^{-4} to 10^{-11} are given.

The effect of very small slopes for lower tolerances could be due to convergence problems of the simplified Newton iteration for relatively large possible step sizes for the low tolerances, see Hairer and Wanner (1999), where it was found that for low tolerances for higher-order methods (R_{ADAU9} and R_{ADAU13}) the CPU-times may even increase. Further, it can be seen that the development of the CPU-times almost follows that of the number of expensive Jacobian matrix evaluations.

5. Conclusions

Several aspects for a flexible MBS description with coupled rigid MBS substructures and flexible bodies described by nonlinear Finite Elements were discussed.

The substructures are interconnected by nonlinear algebraic constraint equations. For the observed DAE of differentiation index 2 complicated partial derivatives of the constraints are required. Handcoding of analytically determined derivatives was compared to an approach utilizing algorithmic differentiation. While AD pro-

duces very reliable results with little human effort, the AD code is considerably slower than that received from manual coding, which is however error-prone. Therefore, AD is applied for verification of the handcoded derivatives which are then utilized in our program system.

The system's governing equations are given in differential algebraic form. The performance of three sophisticated DAE solvers : R_{ADAU5}, D_{DASSL} and M_{EXAX} was investigated regarding their suitability for the application to the usually highly stiff DAE. For this purpose also coordinate projection and Gear-Gupta-Leimkuhler stabilization techniques were implemented for R_{ADAU5} and D_{DASSL} (M_{EXAX} uses an internal projection approach).

Recapitulating it can be said that all solvers show good results regarding accuracy and stability. The A-stable, implicit methods R_{ADAU5} and D_{DASSL} (with maximal order limited to $p_{max}=2$) are superior to the semi-explicit solver M_{EXAX} in terms of the computation times for highly stiff systems. Again, it must be said that M_{EXAX} has not been developed for stiff systems and was only applied to the stiff flexible MBS as we had very promising results for moderately stiff rigid MBS. In this context it would be very interesting to utilize an extrapolation approach with an underlying implicit method.

References

Arnold, M., 1998, "Zur Theorie und zur Lösung von Anfangswertproblemen fur Differentiell-Algebraische Systeme von Hoherem Index," *VDI-Fortschritt-Berichte, Series* 20, No. 264, Dusseldorf : VDI-Verlag.

Bathe, K.-J., 1995, *Finite Element Procedures*, Englewood Cliffs : Prentice Hall.

Bischof, C., Roh, L. and Mauer, A., 1997, "ADIC : An Extensible Automatic Differentiation Tool for ANSI-C," *Software-Practice and Experience*, 27(12), pp. 1427~1456.

Bischof, C. and Bucker, H., 2000, "Computing Derivatives of Computer Programs, In : Modern Methods and Algorithms of Quantum Chemistry," *Proceedings, J. Grotendorst (editor)*, Julich : NIC Series, pp. 315~327.

Brenan, K., Campbell, S. and Petzold, L., 1989, *The Numerical Solution of Initial Value Problems in Ordinary Differential-Algebraic Equations*, New York : North Holland.

Eberhard, P., 2000, *Kontaktuntersuchungen Durch Hybride Mehrkorpersystem/Finite Elemente Simulationen*, Aachen : Shaker Verlag.

Eich-Soellner, E. and Fuhrer, C., 1998, *Numerical Methods in Multibody Dynamics*, Stuttgart : B.G. Teubner.

Gear, C., Leimkuhler, B. and Gupta, G., 1985, "Automatic Integration of Euler-Lagrange Equations with Constraints," *Journal of Computational and Applied Mathematics*, 12(13), pp. 77~90.

Griewank, A., 2000, *Evaluating Derivatives : Principles and Techniques of Algorithmic Differentiation*, Philadelphia : SIAM.

Hairer, E. and Wanner, G., 1996, *Solving Ordinary Differential Equations II*, Berlin : Springer.

Hairer, E. and Wanner, G., 1999, "Stiff Differential Equations Solved by Radau Methods," *Journal of Computational and Applied Mathematics*, 111, pp. 93~111.

Hairer, E., Lubich, C. and Roche, M., 1989, "The Numerical Solution of Differential-Algebraic Systems by Runge-Kutta Methods," *Lecture Notes in Mathematics*, Vol. 1409, Berlin : Springer.

Kubler, L., Eberhard, P. and Geisler, J., 2003, "Flexible Multibody Systems with Large Deformations and Nonlinear Structural Damping using Absolute Nodal Coordinates," *Nonlinear Dynamics*, 34, pp. 31~52.

Kübler, L., Eberhard, P. and Geisler, J., 2003, "Flexible Multibody Systems with Large Deformations Using Absolute Nodal Coordinates for Isoparametric Solid Brick Elements," *Proceedings of the Design Engineering Technical Conferences & Computers and Information in Engineering Conference DETC 2003*, September 2-6, Chicago, USA.

Kübler, L. and Eberhard, P., 2002, "A Formulation for Flexible Multibody Systems with Mixed Cartesian and Relative Coordinates," In : *Proceedings of the NATO Advanced Study Insti-*

tute on Virtual Nonlinear Multibody Systems by Schiehlen, W. and Valasek, M. (Eds.), June 23-July 3, Prague, pp. 113~120.

Lubich, C., 1991, "Extrapolation Integrators for Constrained Multibody Systems," *Impact Comp. Sci. Eng.*, 3, pp. 213~234.

Lubich, C., Nowak, U., Pohle, U. and Engstler, C., 1995, "Numerical Integration of Constrained Mechanical Systems using MEXX," *Mech. Struct. & Mach.*, 23, pp. 473~495.

Nikravesh, P., 1988, *Computer-Aided Analysis of Mechanical Systems*, Englewood Cliffs: Prentice Hall.

Schiehlen, W. and Eberhard, P., 2004, *Technische Dynamik*, Wiesbaden: B.G. Teubner.

Serban, R. and Haug, E., 1988, "Analytical Derivatives for Multibody System Analysis," *Mechanics of Structures and Machines*, 26, pp. 145~173.

Shabana, A., 1998, *Dynamics of Multibody Systems*, Cambridge: University Press.

Simeon, B., 1994, *Numerische Integration Mechanischer Mehrkorpersysteme : Projizierende Deskriptorformen, Algorithmen und Rechenprogramme*, VDI-Fortschritt-Berichte, Series 20, No. 130 Dusseldorf: VDI-Verlag.

Wriggers, P., 2001, *Nichtlineare Finite-Element-Methoden*, Berlin: Springer.