Sensitivity analysis of deployable flexible space structures with a large number of design parameters

Shuai Wang · Qiang Tian · Haiyan Hu · Junwei Shi · Lingbin Zeng

Received: 27 March 2021 / Accepted: 14 July 2021 / Published online: 2 August 2021
© The Author(s), under exclusive licence to Springer Nature B.V. 2021

Abstract The reliability and dynamic performance of deployable flexible space structures significantly depend on their key design parameters. Sensitivity analysis of these design parameters in the frame of multibody dynamics can serve as a powerful tool to evaluate and improve the dynamic performances of deployable space structures. Nevertheless, previous studies on the sensitivity analysis are mainly confined to planar multibody systems with a few design parameters. In this study, an efficient computational methodology is proposed to perform the sensitivity analysis of the complex deployable space structures with a large number of design parameters. Firstly, the analytical sensitivity analysis formulations of objective functions with the parameter-dependent integration bounds are deduced via the direct differentiation method and adjoint variable method. A checkpointing scheme is further introduced to assist the backward integration of the high-dimensional differential algebraic equations of the adjoint variables. The flexible beams in the deployable flexible space structure are described by the locking-free three-node spatial beam elements of absolute nodal coordinate formulation. Furthermore, a parallelized automatic differentiation algorithm is proposed to efficiently evaluate the complex partial derivatives in the sensitivity analysis formulations. Finally, four numerical examples are provided to validate the accuracy and efficiency of the proposed computational methodology.

Keywords Sensitivity analysis · Direct differentiation method · Adjoint variable method · Absolute nodal coordinate formulation · Parallelized automatic differentiation

1 Introduction

The dynamic design of a large-scale deployable space structure is of paramount importance for a successful spacecraft mission. Due to the rapid development of both theories of multibody dynamics and computation technologies, the dynamics of such practical complex
multibody systems (MBSs) can be accurately simulated [1–4]. However, the dynamic simulations merely reveal the dynamic behaviors of the present designs. These designs are usually not the optimal ones. Therefore, further explorations of the design space based on sensitivity analysis (SA) [5–10] are crucial and necessary.

The SA bridges the gap between the dynamic simulations and the gradient-based optimizations [11–13], optimal controls [14] and parametric identifications [15, 16]. SA can be used to obtain the sensitivities of the objective functions with respect to the design parameters. The obtained objective sensitivities quantify the potential improvements of the dynamic performances of a system when certain design parameters are tweaked. Nevertheless, very few works on the SA of the flexible MBSs with a large number of design parameters are reported, since the computation efficiency is the prime bottleneck.

The simplest and most intuitive approach to perform the SA of a multibody system is the finite difference method (FDM) [17]. The implementation of the FDM includes two rounds of dynamic simulations performed by using the nominal parameters and perturbed parameters, respectively. Then, the variations in the objective functions are computed and divided by the parametric difference. While simple to implement, classical FDMs suffer from truncation and cancellation errors. Due to cancellation errors, an excessively small parametric step may lead to totally wrong objective sensitivities. Thus, the SA based on analytical formulations is a promising alternative due to its superior accuracy and efficiency. The direct differentiation method [6–8] and the adjoint variable method [6, 7, 18] are the two alternative approaches for deriving the governing equations in an analytical form for SA.

The direct differentiation method has been widely used because of the easiness in implementation. This approach directly differentiates the equations of motion (EOM) with respect to each design parameter, producing the corresponding equations of sensitivity (EOS). The EOS can be further discretized and integrated in a forward manner along with the EOM. The generalized state sensitivities (the partial derivatives of position, velocity, acceleration and Lagrange multipliers with respect to design parameters) can be determined from the time integration of EOS. Then, the design sensitivities of the objective functions can be determined from the generalized state sensitivities by using numerical quadrature methods [17]. To minimize extra overhead [6], consistent formulations and time integrators can be used to concurrently solve the EOS and EOM due to their inherent relationship. In the computation process, however, the EOM must be differentiated with respect to each parameter separately, and each differentiation will produce a new set of EOS. Thus, this method is quite inefficient when a large number of design parameters are studied.

Dopico et al. [6] proposed a direct differentiation method based on the index-3 augmented Lagrangian formulation with velocity and acceleration projections. The feature of their method is the unified handling of both holonomic and nonholonomic constraints. Bhalerao et al. [8] presented a direct differentiation method in a divide-and-conquer framework and a corresponding parallel algorithm with logarithmic complexity. However, the proposed algorithm cannot be directly applied to the structures with complicated topologies, such as dense nested kinematic loops [19]. Neto et al. [9] demonstrated that the direct sensitivity analysis of the flexible multibody systems can be performed by using the mode component synthesis. Callejo et al. [20] presented a simple vehicle dynamics benchmark for sensitivity analysis based on a state-space formulation. However, the case studies showed that the above works are confined to the dynamic systems with simple topologies and a small number of design parameters.

Compared with the direct differentiation method, the adjoint variable method is more suitable for the systems with relatively few objective functions and a large number of design parameters [21, 22]. The adjoint variable method enables one to evaluate the objective sensitivities directly and avoids the calculation of generalized state sensitivities by augmenting zero terms to the objective sensitivities. The governing equations of the adjoint variable method depend on the introduced adjoint variables instead of the generalized state sensitivities. The objective sensitivities with respect to different design parameters can be computed via the common adjoint variables. However, the governing differential equations in terms of the adjoint variables need to be integrated in backward direction. Such a feature indicates that before the backward integration starts, a large amount of extra computer memory has to be dedicated to store all the state variables throughout the forward integration of the
EOM. Although the checkpointing technique [23, 24] can be utilized to reduce the heavy cost on computer memory, the adjoint method still consumes more memory than the direct differentiation method.

The adjoint variable method can be further categorized as the continuous adjoint variable method (CAVM) and the discrete adjoint variable method (DAVM) [25]. In the CAVM, the continuous EOM are differentiated prior to the temporal discretization of the time integrator. This method requires additional interpolations to obtain the state variables at the integration instants specified by backward integrators, because these instants are usually not coincident with those specified by the forward time integrator. Yet, the DAVM proceeds in a discretize-then-differentiate manner. Thus, extra interpolations can be avoided, because the backward integration hits exactly the same instants of the forward integration. Ebrahimi et al. [12] used the DAVM with a geometric variational integrator for the sensitivity analysis. Schaffer [18] developed a novel piecewise adjoint method by transforming the initial value problem into a boundary value problem, which is well suited for coarse-grained parallel computation. Pi et al. [26] performed the first-order adjoint sensitivity analysis of a flexible crank-slider described by fully parametrized beam elements of the absolute nodal coordinate formulation (ANCF). However, according to the work by Bauchau et al. [27], the ANCF beam element used in the above study suffered from locking problems. Serban et al. [28] creatively extended the adjoint sensitivity analysis to hybrid dynamic systems with memory. Corner et al. [29] used a sensitivity jump matrix to address the discontinuous events in the adjoint sensitivity analysis. Sonneviele et al. [30] reformulated the adjoint variable method in the frame of Lie group, which results in a condensed form of the equations. Lauß et al. [31] applied the discrete adjoint method to the identification of system parameters via the measured accelerations. Nejat et al. [32] adopted the adjoint variable method to compute the structural sensitivity of a flexible crank-slider mechanism described by the approach of floating frame of reference. Wang et al. [33] conducted the adjoint sensitivity analysis based on the geometrically exact beam elements and demonstrated the efficiency of the method through an optimization problem.

The aforementioned direct and adjoint formulations shed light on how to compute the objective sensitivities. In each time integration step of the above formulations, quite a few complex partial derivatives [18, 34] are required for the design sensitivities of the mass matrix, the Jacobian and design sensitivities of the elastic forces of finite elements [35]. If they were not computed efficiently and accurately, the performance of the SA algorithm would not be successful. There are four ways [36] to compute derivatives: finite difference, analytical differentiation, symbolic differentiation and automatic differentiation (AD). The real-step finite difference method [17] used for computing derivatives still suffers from the truncation and cancellation errors, thus the perturbation step size is strictly restricted, and a trial-and-error process is also required. The analytical differentiation method is accurate and efficient if the closed-form formulae can be derived and correctly implemented in the program. However, the derivation of analytical derivatives is a usually cumbersome or even impossible task sometimes, especially for the complex systems with a large number of design parameters. Symbolic differentiation can alleviate some of the burden of deriving analytical expressions for the partial derivatives, but suffers from the expression swell issues [37]. The automatic differentiation method, also called the algorithmic differentiation method, is a promising alternative compared with the above three methods. The AD method can get machine-precision derivatives efficiently with moderate efforts [38]. The method computes the derivatives by applying the chain rule of derivative calculus repeatedly to the elementary arithmetic operations in the program, without the incurring of truncation errors. The AD method can be performed in two ways, the operator overloading [39–41] and the source-to-source transformation [42, 43], while each way also covers two modes, that is, the forward mode and the reverse mode [38]. Recent years have seen successful applications of AD to evaluate derivatives in the dynamic simulation of multibody systems [44, 45], but only for multi-rigid body systems. In this study, the complex partial derivatives are determined from the forward mode AD based on operator overloading.

The remaining parts of the paper are organized as follows. In Sect. 2, the analytical formulations of SA are deduced via the direct differentiation method and the adjoint variable method, respectively. In this study, the upper and lower integration bounds of the objective function also depend on the design parameters. In
Sect. 3, to compute the complex derivatives of the design sensitivities and Jacobian of the nonlinear elastic forces of a beam element efficiently and accurately, a parallelized automatic differentiation algorithm is developed by using OpenMP directives. In Sect. 4, three numerical examples are firstly provided to validate the proposed formulations. The fourth example is to demonstrate that the proposed computation methodology can be used to the SA of deployable flexible space structures with a large number of design parameters. Finally, the main conclusions are drawn and the perspectives for future researches are outlined in Sect. 5.

2 Analytic sensitivity formulations for a flexible multibody system

2.1 Dynamic equations of a flexible multibody system

The dynamic equations for a flexible multibody system with $m$ holonomic constraints described by $n$ generalized coordinates can be written as a set of differential algebraic equations (DAEs) of index-3 as follows

\[
\begin{align*}
\mathbf{M}(\mathbf{a})\ddot{\mathbf{q}}(\mathbf{a}) + \mathbf{D}(\mathbf{q}, \mathbf{\dot{q}})\dot{\mathbf{\lambda}}(\mathbf{a}) &= \mathbf{F}(\mathbf{q}(\mathbf{a}), \mathbf{\dot{q}}(\mathbf{a})), \\
\mathbf{\Phi}(\mathbf{q}(\mathbf{a}), t) &= \mathbf{0}
\end{align*}
\]

(1)

where $\mathbf{a} \in \mathbb{R}^p$ denotes the design parameter vector, $\mathbf{M} \in \mathbb{R}^{m \times n}$ is the mass matrix, $\mathbf{q}$ and $\dot{\mathbf{q}} \in \mathbb{R}^n$ denote the vectors of generalized coordinates, velocities and accelerations, respectively, $\mathbf{\lambda} \in \mathbb{R}^m$ is the vector of Lagrange multipliers, $\mathbf{\Phi} \in \mathbb{R}^m$ is the vector of constraints, $\mathbf{\Phi}_\mathbf{q} \in \mathbb{R}^{m \times n}$ is the Jacobian of vector $\mathbf{\Phi}$, and $\mathbf{F}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^n$ is the vector of resultant generalized forces, including vector $\mathbf{F}^{\text{int}}(\mathbf{q}) \in \mathbb{R}^n$ for internal elastic forces and vector $\mathbf{F}^{\text{ext}}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^n$ for external applied forces, namely

\[
\mathbf{F}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{F}^{\text{int}}(\mathbf{q}) + \mathbf{F}^{\text{ext}}(\mathbf{q}, \dot{\mathbf{q}}),
\]

(2)

where $\mathbf{F}^{\text{int}}(\mathbf{q})$ is obtained by assembling the elastic forces of all the elements, and $\mathbf{F}^{\text{ext}}(\mathbf{q}, \dot{\mathbf{q}})$ is mainly contributed by the actuator forces considering energy dissipations.

2.2 SA based on the direct differentiation method

Let the dynamic response of concern be from time $t_0$ to $t_f$, a general objective function that depends on the design parameters and state variables can be written as

\[
\psi = [w(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \lambda, \mathbf{a})]_{t_f} + \int_{t_0}^{t_f} g(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \lambda, \mathbf{a}) \, dt,
\]

(3)

where $[\cdot]_{t_0}$ and $[\cdot]_{t_f}$, respectively, indicate that the values of the term in the brackets are evaluated at the moment $t_0$ and $t_f$, the scalar $w$ depends on the generalized state variables of the system at the moment $t$, the integrand $g$ depends on the dynamic outputs over the time interval $[t_0, t_f]$, and can be cast as a nested space-dependent integral. Different from most previous studies [5–9], the lower bound $t_0$ and upper bound $t_f$ of integration time in Eq. (3) are also functions of design parameter vector $\mathbf{a}$, and they can be obtained by solving the equations

\[
\begin{align*}
\Gamma^1(\mathbf{q}(\mathbf{a}, t_0), \dot{\mathbf{q}}(\mathbf{a}, t_0), \ddot{\mathbf{q}}(\mathbf{a}, t_0), t_0, \mathbf{a}) &= 0, \\
\Gamma^2(\mathbf{q}(\mathbf{a}, t_f), \dot{\mathbf{q}}(\mathbf{a}, t_f), \ddot{\mathbf{q}}(\mathbf{a}, t_f), t_f, \mathbf{a}) &= 0,
\end{align*}
\]

(4)

where $\Gamma_1$ and $\Gamma_2$ are called the parameter-dependent kinematic criteria. Equation (4) can be used for the space mechanisms whose deployment time depends on the design parameters, such as the self-actuating and self-locking mechanisms [46].

Then, according to the direct differentiation method [5], the objective sensitivity can be evaluated by differentiating Eq. (3) with respect to $\mathbf{a}$, as

\[
\nabla_a \psi = [w_\mathbf{q} \partial_\mathbf{q} + w_\dot{\mathbf{q}} \partial_\dot{\mathbf{q}} + w_\ddot{\mathbf{q}} \partial_{\ddot{\mathbf{q}}} + w_\lambda \partial_\lambda + w_a]_{t_f}
\]

\[
+ \int_{t_0}^{t_f} (g_\mathbf{q} \partial_\mathbf{q} + g_\dot{\mathbf{q}} \partial_\dot{\mathbf{q}} + g_\ddot{\mathbf{q}} \partial_{\ddot{\mathbf{q}}} + g_\lambda \partial_\lambda + g_a) \, dt
\]

\[
+ \frac{df_r}{da} \cdot [g]_{t_f} - \frac{df_0}{da} \cdot [g]_{t_0},
\]

(5)

where $\nabla$ is the gradient operator, $\partial_\mathbf{q}$, $\partial_\dot{\mathbf{q}}$, $\partial_{\ddot{\mathbf{q}}}$, $\partial_\lambda$, and $\partial_a$ are, respectively, the partial derivatives of $\mathbf{q}$, $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$, and $\lambda$ with respect to $\mathbf{a}$. It is worthy to point out that Eq. (5) is different from Eq. (1.14) in the reference [5] because the lower bound $t_0$ and upper bound $t_f$ here are also functions of the parameter vector $\mathbf{a}$. It is straightforward to compute the terms $\frac{df_0}{da}$ and $\frac{df_r}{da}$ in Eq. (5) by
differentiating Eq. (4) with respect to parameter vector $\mathbf{a}$ as follows

$$\begin{align*}
\frac{d\mathbf{f}_0}{da} &= \left[ -\Gamma_q^T \mathbf{q}_a - \Gamma_q^T \dot{\mathbf{q}}_a - \Gamma_q^T \ddot{\mathbf{q}}_a - \Gamma_q^T \dddot{\mathbf{q}}_a \right]_{t_0} \\
\frac{d\mathbf{f}_F}{da} &= \left[ -\Gamma_q^T \mathbf{q}_a - \Gamma_q^T \dot{\mathbf{q}}_a - \Gamma_q^T \ddot{\mathbf{q}}_a - \Gamma_q^T \dddot{\mathbf{q}}_a \right]_{t_F}.
\end{align*}$$  \quad (6)

As shown in Eq. (5), the evaluation of the objective sensitivity requires the results of the generalized state sensitivities $(\mathbf{q}_a, \dot{\mathbf{q}}_a, \ddot{\mathbf{q}}_a, \lambda_a)$. They can be obtained via the direct differentiation of Eq. (1) with respect to vector $\mathbf{a}$, and then the governing EOM of all the design parameters can be written as

$$\begin{align*}
\left\{ \begin{array}{l}
\mathbf{M}_a \ddot{\mathbf{q}}_a - \mathbf{F}_a \mathbf{q}_a + \left( \mathbf{M}_a \lambda_a + \mathbf{F}_a^T \lambda_a \right) \mathbf{q}_a + \mathbf{F}_a \lambda_a = \mathbf{F}_a - \mathbf{M}_a \lambda_a - \mathbf{F}_a \lambda_a,

\mathbf{q}_a(t) = -\mathbf{F}_a(t)
\end{array} \right.
\end{align*}$$  \quad (7)

where

$$\mathbf{F}_a = -\frac{\partial \mathbf{F}(\mathbf{q}, \dot{\mathbf{q}})}{\partial \mathbf{q}}, \quad \mathbf{F}_q = -\frac{\partial \mathbf{F}(\mathbf{q}, \dot{\mathbf{q}})}{\partial \mathbf{q}}.$$  \quad (8)

2.3 SA based on the adjoint variable method

In the above direct differentiation method, the generalized state sensitivities with respect to each design parameter have to be computed separately. This way is especially inefficient for the systems with a large number of design parameters. According to the adjoint variable method [28, 30–33], the computations of generalized state sensitivities can be circumvented by a clever augment of zero terms in the objective sensitivities, and only one set of DAEs is required to be solved.

Using the method of integration by parts, Eq. (5) can be further cast as

$$\nabla_a \psi = \left[ w_q \mathbf{q}_a + w_q \dot{\mathbf{q}}_a + w_q \ddot{\mathbf{q}}_a + w_q \dddot{\mathbf{q}}_a \right]_{t_F} + \left[ \left( \mathbf{g}_q - \frac{d \mathbf{g}_q}{dt} \right) \mathbf{q}_a + \mathbf{g}_q \dot{\mathbf{q}}_a + \mathbf{g}_q \ddot{\mathbf{q}}_a + \mathbf{g}_q \dddot{\mathbf{q}}_a \right]_{t_0} + \int_{t_0}^{t_F} \left( \left( \mathbf{g}_q - \frac{d \mathbf{g}_q}{dt} + \frac{d^2 \mathbf{g}_q}{dt^2} \right) \mathbf{q}_a + \mathbf{g}_q \lambda_a + \mathbf{g}_a \right) dt,$$

where $\| \mathbf{f}_0 \| = \| \mathbf{f}_F \|$. From the adjoint method [5, 21], the time integration of the sensitivity of the first equation in Eq. (1) can be written as

$$\int_{t_0}^{t_F} \mathbf{f}^T (\mathbf{M}_a \dot{\mathbf{q}}_a + \mathbf{F}_a^T \lambda_a - \mathbf{F}_a) \frac{d\mathbf{a}}{dt} = 0,$$

(10)

where $\mathbf{f}$ is the adjoint vector defined for the dynamic equations. Also, by using the method of integration by parts, Eq. (10) can be further cast as

$$\left[ \mathbf{f}^T (\mathbf{M}_a \dot{\mathbf{q}}_a + \mathbf{F}_a^T \lambda_a - \mathbf{F}_a) \right]_{t_0} + \int_{t_0}^{t_F} \left( \mathbf{f}^T (\mathbf{M}_a \dot{\mathbf{q}}_a + \mathbf{F}_a^T \lambda_a - \mathbf{F}_a) \right) d\mathbf{a} = 0.$$

(11)

Similarly, the time integration of the sensitivity of the second equation in Eq. (1) can be further written as

$$\int_{t_0}^{t_F} \mathbf{v}^T \frac{d\mathbf{f}_F}{d\mathbf{a}} d\mathbf{a} = \int_{t_0}^{t_F} \mathbf{v}^T (\mathbf{M}_a \dot{\mathbf{q}}_a + \mathbf{F}_a) d\mathbf{a} = 0$$

(12)

where $\mathbf{v}$ is the adjoint vector defined for the constraint equations.

Let Eq. (4) be differentiated with respect to vector $\mathbf{a}$ and then, respectively, multiplied by the introduced adjoint scalars $\lambda_1$ and $\lambda_2$, the following adjoint equations can be established as

$$\begin{align*}
\lambda_1 \left[ \Gamma_q^T \mathbf{q}_a + \Gamma_q^T \dot{\mathbf{q}}_a + \Gamma_q^T \ddot{\mathbf{q}}_a + \Gamma_q^T \dddot{\mathbf{q}}_a \right]_{t_0} + \lambda_1 \left[ \Gamma_q^T \mathbf{q}_a + \Gamma_q^T \dot{\mathbf{q}}_a + \Gamma_q^T \ddot{\mathbf{q}}_a + \Gamma_q^T \dddot{\mathbf{q}}_a \right]_{t_F} &= 0, \\
\lambda_2 \left[ \Gamma_q^T \mathbf{q}_a + \Gamma_q^T \dot{\mathbf{q}}_a + \Gamma_q^T \ddot{\mathbf{q}}_a + \Gamma_q^T \dddot{\mathbf{q}}_a \right]_{t_0} + \lambda_2 \left[ \Gamma_q^T \mathbf{q}_a + \Gamma_q^T \dot{\mathbf{q}}_a + \Gamma_q^T \ddot{\mathbf{q}}_a + \Gamma_q^T \dddot{\mathbf{q}}_a \right]_{t_F} &= 0,
\end{align*}$$

(13)

Subtracting Eqs. (11), (12) and (13) from Eq. (9), the augmented equation of the objective sensitivity can be written as

$$\nabla_a \psi = \left[ w_q \mathbf{q}_a + w_q \dot{\mathbf{q}}_a + w_q \ddot{\mathbf{q}}_a + w_q \dddot{\mathbf{q}}_a \right]_{t_F} + \left[ \left( \mathbf{g}_q - \frac{d \mathbf{g}_q}{dt} \right) \mathbf{q}_a + \mathbf{g}_q \dot{\mathbf{q}}_a + \mathbf{g}_q \ddot{\mathbf{q}}_a + \mathbf{g}_q \dddot{\mathbf{q}}_a \right]_{t_0} + \int_{t_0}^{t_F} \left( \left( \mathbf{g}_q - \frac{d \mathbf{g}_q}{dt} + \frac{d^2 \mathbf{g}_q}{dt^2} \right) \mathbf{q}_a + \mathbf{g}_q \lambda_a + \mathbf{g}_a \right) dt.$$
To circumvent the computation of state sensitivities, the coefficient expressions of $q_a$ and $\lambda_a$ in the integrands of Eq. (14) are set to zero. Then, an index-3 backward time integration. The final values of vectors $[\mathbf{\hat{q}}_a]_{t_F}$, $[\mathbf{\hat{\lambda}}_a]_{t_F}$, and $\frac{d\mathbf{q}}{da}$ in Eq. (14) to zero, and evaluating the second equation in Eq. (15) at moment $t_F$, the equations for solving $[\mathbf{\hat{\mu}}]_{t_F}$, $[\mathbf{\hat{v}}]_{t_F}$, $\chi_2$ can be written as

$$\begin{align*}
\left[\mathbf{M}^T \mathbf{\mu} + (\mathbf{M} + \mathbf{F}_q)^T \mathbf{\hat{\mu}} + \mathbf{F} \mathbf{\hat{\mu}} - \chi_2 \mathbf{\hat{\lambda}} \right]_{t_F} &= \left[\frac{d\mathbf{q}}{dt} - \mathbf{g}_q - \mathbf{w}_q + \chi_2 \mathbf{\hat{\lambda}} \right]_{t_F}, \\
\left[\mathbf{M}^T \mathbf{v} + (2\mathbf{M} + \mathbf{F}_q)^T \mathbf{\hat{v}} + \mathbf{F} \mathbf{\hat{v}} - \chi_2 \mathbf{\hat{\lambda}} \right]_{t_F} &= \left[g_q + \mathbf{w}_q - \chi_2 \mathbf{\hat{\lambda}} \right]_{t_F},
\end{align*}$$

(16a)

(16b)

(16c)

(16d)

(16e)

(16f)

It can be found that Eqs. (16c) and (16f) fail to give a unique solution of the adjoint variable $\chi_2$. To solve this problem, the following conditions should be further satisfied

$$\left[\mathbf{\hat{\Gamma}}_q \right]_{t_F} = 0, \quad [w_q]_{t_F} = 0.$$  

(17)

Again, it can be found that Eqs. (16b) and (16f) are not compatible either. Thus, the following zero terms evaluated at moment $t_F$ are augmented to Eq. (14),

$$\begin{align*}
\left[\frac{\mathbf{\hat{\mu}}}{\mathbf{\hat{v}}} \right]_{t_F} &= \left[\frac{\mathbf{\hat{\mu}}}{\mathbf{\hat{v}}} \right]_{t_F} - \frac{\mathbf{d}\mathbf{q}}{da} + \mathbf{g}_q, \\
\left[\mathbf{\hat{\mu}} \right]_{t_F} &= \left[\mathbf{\hat{\mu}} \right]_{t_F} + \frac{\mathbf{d}\mathbf{q}}{da} + \mathbf{g}_q.
\end{align*}$$

(18)

where the new adjoint vectors $\gamma$ and $\eta$ are introduced to multiply the design sensitivities of the position-level and velocity-level constraint equations, respectively. After adding Eq. (18) to Eq. (14), the augmented objective sensitivity can be recast as
\[
\n\nabla \psi = \left[ (w_q + g_q - \frac{dg_q}{dr} + (\dot{M} + \dot{F}_q)^T \dot{\mu} + \dot{M}^T \dot{\mu} + \gamma^T \Phi_q + \eta^T (\Phi_{qq} \dot{q} + \Phi_q) - \chi_2 \Gamma_{\dot{q}}^2) q_a + [w_q - \chi_2 \Gamma_{\dot{q}}^2]_{\dot{q}a} \right]_{\dot{t}_f} + \left[ (w_q + g_q - \dot{M}^T \mu + \mu^T (F_q - M) - \chi_1 \Gamma_{\mu}^1) q_a + (\mu^T M - g_q - \chi_1 \Gamma_{\mu}^1) q_a - \chi_1 \Gamma_{\mu}^1]_{\dot{q}a} \right]_{\dot{t}_f}
\]

Then, the compatible equations can be established and solved for starting the backward integration of Eq. (15) as follows

\[
\begin{align*}
&M_T \mu + (M + F_q)^T \mu + \Phi_q^T \gamma + (\Phi_{qq} \dot{q} + \Phi_q)^T \eta = [dg_q \overline{dr} - g_q - w_q + \chi_2 \Gamma_{\dot{q}}^2]_{\dot{t}_f} \\
&[\Gamma^2 \chi_2]_{\dot{t}_f} = [g]_{\dot{t}_f} \\
&[\Phi_q]_{\dot{t}_f} = [g \chi_2]_{\dot{t}_f} \\
&[\Phi_{qq} \dot{q} + \Phi_q]_{\dot{t}_f} = [dg_q \overline{dr}]_{\dot{t}_f}
\end{align*}
\]

where the final values of \( [\mu]_{\dot{t}_f}, [\dot{\mu}]_{\dot{t}_f}, [\gamma]_{\dot{t}_f}, [\eta]_{\dot{t}_f} \) and \( \chi_2 \) are solved.

Finally, according to Eq. (15) and Eq. (20), Eq. (19) can be simplified as

\[
\nabla \psi = \int_{t_0}^{t_f} \left( [\dot{M} \dot{q} + \Phi_q \dot{\lambda} - F_q - v^T \Phi_q] dt + \int_{t_0}^{t_f} \left( [\dot{M}^T \dot{\mu} + \mu^T (\dot{F}_q - \dot{M}) - \chi_2 \dot{\Gamma}_{\dot{q}}^2] \right)_{\dot{t}_f} \right)
\]

where \( \chi_1 \) is obtained by solving \( [\dot{\Gamma}^1]_{\dot{q}a} \chi_1 = [-g]_{\dot{t}_f} \). It is worth noting that the simplified objective sensitivities with respect to different design parameters can be computed via the shared adjoint vectors \( \mu \) and \( \nu \) determined from Eq. (15). Thus, the adjoint variable method is favorable for the SA of system with a large number of design parameters.

2.4 Time integration and checkpointing scheme

For the index-3 DAEs formulated by Eq. (1), the implicit Newmark integrator described in [47] is frequently used due to its simplicity and wide applicability. The use of other time integrators to discretize the DAEs can also be found in the literature works, such as the geometric variational integrator [12], the HHT integrator [22] and the diagonally implicit Runge–Kutta integrator [48]. Taking the time integration of Eq. (7), for example, the position sensitivities \( \dot{q}_a \) and \( \dot{q}_{a+1} \) of the time step \( k + 1 \) are treated as the unsolved primary variables. Then, the velocity sensitivities \( \dot{q}_a^{k+1} \) and acceleration sensitivities \( \dot{q}_{a+1}^{k+1} \), respectively, read

\[
\begin{align*}
\dot{q}_a^{k+1} &= \frac{h}{\beta} \dot{q}_a^{k} + \dot{q}_a^{k} = -\frac{h}{\beta} \dot{q}_a^{k} - \left( \frac{7}{2\beta} - 1 \right) \dot{q}_a^{k}, \\
\dot{q}_{a+1}^{k+1} &= -\frac{h}{\beta} \dot{q}_{a+1}^{k} - \left( \frac{7}{2\beta} - 1 \right) \dot{q}_{a+1}^{k} \\
\dot{q}_a^{k+1} &= -\frac{h}{\beta} \dot{q}_a^{k} - \frac{h}{\beta} \dot{q}_{a+1}^{k} - \left( \frac{7}{2\beta} - 1 \right) \dot{q}_a^{k}, \\
\dot{q}_{a+1}^{k+1} &= -\frac{h}{\beta} \dot{q}_{a+1}^{k} - \left( \frac{7}{2\beta} - 1 \right) \dot{q}_{a+1}^{k}.
\end{align*}
\]

where \( h \) denotes the time step, \( \beta \) and \( \gamma \) are scalar parameters that affect the precision and stability of the integrator. This method is implicit and A-stable provided that \( \gamma \geq 1/2 \) and \( \beta \geq (\gamma+1/2)^2/4 \) hold [49]. In the case of the backward integration required by the
adjoint index-3 DAEs, the time step $h$ in Eq. (22) should be substituted with $-h$.

Applying the above integration rules to the temporal discretization of Eq. (7) at time step $k + 1$ leads to

$$
\begin{array}{c}
\left[ M + \beta h^2 \left( Mq \dot{q} + \Phi_{qq}^T \lambda - F_q \right) - \gamma h F_q \Phi_q^T \right] \dot{q}^{k+1} + \frac{q_q}{C_0} M_0^{k+1} \\
\Phi_q \\
0
\end{array}
= \left[ \beta h^2 \left( F_a - Mq \ddot{q} - \Phi_{qq}^T \dot{q} - Mq \dot{q}^T + F_q \right) \right]^{k+1} - \Phi_a^{k+1}
$$

(23)

This equation is a linear one in terms of the generalized state sensitivities $q_a$ and $\lambda_a$. The leading matrix of the left-hand side in Eq. (23) is identical to the one used in the dynamic simulation. Thus, efficiency can be improved if consistent formulations are utilized for both the dynamic simulation and sensitivity analysis.

Equation (15) of the adjoint variable method is solved backward in integration time, and the backward solution process can only be performed after the forward time integration of dynamic equations is finished. To proceed the backward integration, the generalized state variables at the desired instants are required. One simple strategy is to store all the generalized state variables at the discretized time steps during the forward integration. However, this way is only acceptable for small-scale problems due to the limited memory of a computer. The checkpointing scheme proposed in [23, 24] is a compromise between the storage space and the computation efficiency. Figure 1 schematically shows the implementation of this scheme, which can be summarized as the following six steps:

1. Perform the forward integration of the EOM step by step ($k \rightarrow k + 1$);
2. Dump the generalized state vectors ($q$, $\dot{q}$, $\ddot{q}$ and $\lambda$) at the specified check points;
3. Proceed the forward integration consecutively until the final moment $t_F$;
4. Evaluate the final values of the adjoint variables by solving Eq. (20) before starting the backward integration;
5. Use the stored generalized state vectors in stage 2 to reproduce the results of forward integration of dynamic equations;
6. Propagate the adjoint variables backward until the starting moment $t_0$.

3 Evaluating partial derivatives in the analytical formulations

3.1 Analytical and symbolic differentiations

Analytical differentiation is very efficient and accurate, but only applicable when the closed-form solution can be found. Computing complex derivatives often require much mental work. Symbolic software, such as Maple and Mathematica, can be used to alleviate the deduction work of the analytical differentiation [36]. Yet, one major issue of the symbolic differentiation is the expression swell [37]. The symbolic differentiation is able to handle the control flows in the computation of complex derivatives, such as if-else statements, loops and recursions, but the produced formulae are excessively complicated. In this study, only some simple quantities such as $\Phi_{qq}^T \lambda$ and $\Phi_{qq} \dot{q}$ in Eq. (19) are determined from the
analytical or symbolic differentiation. Also, it is wise to do so because these constraint-related matrices usually have regular sparse structures depending on the categories of the kinematic pairs. Once the analytical formulae of these matrices for a specific kinematic pair are derived, they can be exploited for computing the derivatives efficiently.

3.2 Real-step finite difference method

The finite-difference method (FDM) is the simplest and most intuitive technique for computing numerical derivatives. For the sake of simplicity, only the differentiation of a scalar $f$ with respect to the parameter vector $a \in \mathbb{R}^p$ is considered here. Accordingly, the proposed formulae can be easily extended to the differentiation of vectors, matrixes and high-order tensors.

According to the Taylor series expansion, the first-order derivative of $f$ in the real-step FDM can be approximated as

$$\frac{\partial f(a)}{\partial a_s} = \frac{f(a + \delta a_s \cdot e_s) - f(a)}{\delta a_s}, \quad s = 1, 2, \ldots, p, \quad (24)$$

where $\delta a_s$ is a small perturbation to the $s$-th design parameter of $a$, $e_s \in \mathbb{R}^p$ is the unit vector, the $s$-th entry of which is one and all the other entries are zeros. While simple to implement, the accuracy of the real-step FDM is sensitive to the perturbation step. If the perturbation step is too large, the result is inaccurate due to truncation errors. Otherwise, the result may be totally wrong due to canceling-off errors if the perturbation step is too small. Trial-and-error is the only way to deal with this contradiction. Again, there is no universal criterion to determine the adequate perturbation step for different physical problems. Although the FDM is not used in the codes for computing the derivatives in the analytical formulations of Sect. 2, it is widely used for checking other methods via the adequate perturbation steps.

3.3 Automatic differentiation method

Compared with the above methods, the automatic differentiation method has several advantages for computing derivatives. Firstly, AD is convenient to implement because there is no need to derive the closed-form formulae of the derivatives. Secondly, AD incurs no truncation and cancelling-off errors compared with the FDM. Also, AD gets rid of the trial-and-error process in the real-step FDM and enables one to obtain the derivatives with respect to the design parameters simultaneously and does not need to perturb each parameter separately. However, AD has its own weaknesses. The performances of AD strongly depend on specific implementations. An elegant implementation of AD requires deep insights and testing practices, otherwise, the efficiency of the AD algorithm will be attenuated. Another weakness is that AD cannot output derivatives if one does not have access to the functions differentiated.

The categories of AD can be classified into source-to-source transformation and operator overloading [38]. Source-to-source transformation works as a preprocessor to generate the codes for computing derivatives. The generated codes are usually quite complex, which prohibits an easy integration with the program of analytical SA in Sect. 2. Yet, operator-overloading AD works at run-time by overloading the elementary arithmetic operators and intrinsic functions. The forward mode AD is used in this study because each finite element has a small number of design parameters and the computation of derivatives runs for a single element.

AD computes the numerical derivatives of mathematical expressions by applying the chain rule circularly and automatically. All mathematical formulations, no matter how complex they appear, can be decomposed into a set of variables subject to some elementary arithmetic operations. Thus, the computational graph [38] in the frame of AD can be built to reveal the interconnections of these variables and the sequence of the arithmetic operations. The variables are classified into three categories, namely the independent, dependent and intermediate variables. Independent variables, usually the design parameters for the sensitivity analysis, are those located at the beginning of a computational graph. Dependent variables are the final ones whose derivatives with respect to the independent variables are computed. The intermediate variables are all others in a computational graph, which “carry” the derivatives from the independent variables to the dependent variables.

In the operator-overloading approach, a new data structure of AD [40] is defined to declare the independent, intermediate and dependent variables.
A scalar \(\bar{x}\) is extended to vector \(\mathbf{x}_{\text{AD}}\) as follows

\[
\mathbf{x}_{\text{AD}} = \begin{bmatrix}
\bar{x} & \mathbf{d}_x & \mathbf{n}_x^T
\end{bmatrix}^T,
\]

where \(\bar{x}\) is the function value that depends on the independent variables, \(\mathbf{d}_x \in \mathbb{R}^p\) is the vector of the nonzero partial derivatives of \(\bar{x}\) with respect to different independent variables, \(p\) is the maximal dependency number introduced to store the derivatives in a compressed format, \(\mathbf{n}_x \in \mathbb{N}^{p+1}\) holds the indices of dependencies when the independent variables are numbered as \(1, 2, \ldots, p\), \(p\) is the total number of independent variables. The first entry of \(\mathbf{n}_x\) holds the current number of nonzero derivatives, and the remaining \(p\) entries are the corresponding indices of the independent variables with respect to which the nonzero derivatives in \(\mathbf{d}_x\) are taken.

Figure 2 shows two examples of declaring the independent and intermediate variables using Eq. (25). In Fig. 2, \(\bar{x}_i\) denotes the original scalar format of the \(i\)-th independent variable, and \((\mathbf{x}_i)_{\text{AD}}\) and \((\mathbf{y}_i)_{\text{AD}}\) denote the AD data structure format of the \(i\)-th independent variable and intermediate variable, respectively. Here, the total number of independent variables is \(p = 6\), and \(i = 1, 2, \ldots, p\). The ticks in the grids denote the dependencies between the declared variables of AD data structure and independent variables, which further indicate that the declared variables have nonzero derivatives with respect to the corresponding independent variables.

Due to the locality of calculations, the intermediate variables usually depend only on a few of the independent variables. Thus, one scattered pattern of the distribution of dependencies is assumed in the right side of Fig. 2. Here, the maximal dependency number is \(\bar{p} = 3\) and can be chosen before the compilation of the program \([40]\). Hence, the dimension of \((\mathbf{y}_i)_{\text{AD}}\) is \(2\bar{p} + 2\) rather than \(2p + 2\). It is intuitive that the compressed storage of nonzero derivatives and corresponding indices will increase the efficiency of follow-up computations for overloaded operators.

Using Eq. (25), the \(i\)-th independent variable in the left side of Fig. 2 with the reference function value \(\bar{x}_i\) can be extended as

\[
(\mathbf{x}_i)_{\text{AD}} = \begin{bmatrix}
\bar{x}_i & \mathbf{d}_{x_i}^T & \mathbf{n}_{x_i}^T
\end{bmatrix}^T,
\]

where \(\mathbf{d}_{x_i}^T = [1.0 \ 0.0 \ 0.0 \ 0.0 \ 0.0 \ 0.0]_{1 \times \bar{p}}\) and \(\mathbf{n}_{x_i}^T = [1 \ i \ 0 \ 0 \ 0 \ 0]_{1 \times (p+1)}\). The scalar 1.0 in \(\mathbf{d}_{x_i}^T\) indicates that the only nonzero derivative of \(i\)-th independent variable is 1.0 with respect to itself, while the unused entries in \(\mathbf{d}_{x_i}^T\) are filled in with 0.0 for vector alignment. The 1 and \(i\) in \(\mathbf{n}_{x_i}^T\), respectively, indicate that the \(i\)-th independent variable has one nonzero derivative in total and the corresponding index of that nonzero derivative is \(i\), while the unused entries in \(\mathbf{d}_{x_i}^T\) are also filled in with 0.

According to Eq. (26), the initial function values and derivatives are carried by the independent variables using the data structure of AD. The remaining work of AD is to propagate these derivatives along the computational graph among the intermediate variables till the dependent variables. Such a propagation is achieved by overloading the elementary arithmetic operators and intrinsic functions. The underlying principle of overloading is applying the chain rule of differentiation. As each intermediate variable is also
Table 1 Overloading rules of arithmetic operators and elementary functions

| Arithmetic operator/elementary function | Scalar arithmetic | Overloaded arithmetic |
|-----------------------------------------|-------------------|----------------------|
| ±                                       | \( z = x \pm y \) | \( z^{\text{AD}} = x^{\text{AD}} \pm y^{\text{AD}} \Rightarrow \{ z = x \pm y, d_z = d_x \pm d_y, n_z = n_x \cup n_y \} \) |
| *                                       | \( z = x \ast y \) | \( z^{\text{AD}} = x^{\text{AD}} \ast y^{\text{AD}} \Rightarrow \{ z = x \ast y, d_z = x \cdot d_y + y \cdot d_x, n_z = n_x \cup n_y \} \) |
| /                                       | \( z = x/y \)    | \( z^{\text{AD}} = x^{\text{AD}}/y^{\text{AD}} \Rightarrow \{ z = x/y, d_z = y \cdot d_x - x \cdot d_y, n_z = n_x \cup n_y \} \) |
| sin( )                                  | \( z = \sin(x) \) | \( z^{\text{AD}} = \sin(x^{\text{AD}}) \Rightarrow \{ z = \sin(x), d_z = \cos(x) \cdot d_x, n_z = n_x \cup n_y \} \) |

extended as the new data structure, the overloaded operators and functions need not only to calculate the function values, but also to update the nonzero derivatives and the corresponding indices of dependency.

Table 1 presents the overloading rules of several common operators and intrinsic functions. As shown in the third column of the table, the three parts of the resultant are calculated separately. The first part is to perform the ordinary scalar arithmetic of the expression, the second part is to calculate the nonzero derivatives of the right-hand side based on the chain rule of differentiation and the third part is to update the dependency indices of the right-hand side based on the union of dependency indices of the left-hand sides.

Furthermore, the above-mentioned overloading process covers only the overloaded arithmetic between variables of AD data structure. The mixed mode arithmetic, like multiplication of a scalar and a variable of AD data structure, is not covered but can be accomplished similarly.

To illustrate the usage of the AD data structure and overloaded operators, consider an simple example, the AD of a scalar expression \( z = \bar{x}_2 + 3x_4 + 5\bar{x}_6 \), where \( \bar{x}_2, x_4 \) and \( \bar{x}_6 \) are the independent variables, \( \bar{z} \) is the intermediate variable. The overloaded version of this expression is \( z^{\text{AD}} = (x_2)^{\text{AD}} + 3 \ast (x_4)^{\text{AD}} + 5 \ast (x_6)^{\text{AD}} \). As the addition and multiplication operators are overloaded, the corresponding overloading rules in Table 1 are used such that the resulting intermediate variable in the extended form \( z^{\text{AD}} \) can be written as

\[
z^{\text{AD}} = \left[ \bar{z} \quad d^{\text{T}}_z \quad n^{\text{T}}_z \right]^{\text{T}},
\]

where \( \bar{z} = \bar{x}_2 + 3x_4 + 5\bar{x}_6 \),

\[
d^{\text{T}}_z = \begin{bmatrix} 1.0 & 3.0 & 5.0 & 0.0 & \cdots & 0.0 \end{bmatrix}_{1 \times (p+1)},
\]

\[
n^{\text{T}}_z = \begin{bmatrix} 3 & 2 & 4 & 6 & 0 & \cdots & 0 \end{bmatrix}_{1 \times (p+1)}.
\]

Obviously, the derivative information is automatically transferred from the independent variables \( (x_2)^{\text{AD}}, (x_4)^{\text{AD}} \) and \( (x_6)^{\text{AD}} \) to the intermediate variable \( z^{\text{AD}} \).

It is clearly shown that the AD data structure and overloaded operators enable the propagation of the derivatives along the computational graph automatically [38]. Moreover, the propagation will not be hindered by the control flows, such as loops, recursions, if-else statements and subroutine calls. Thus, the derivatives relay consecutively among independent variables, intermediate variables and dependent variables. The accurate derivatives of the dependent variables with respect to the independent variables can be easily extracted from the new data structure of AD.

3.4 Application of a parallelized AD to a locking-free beam element of ANCF

In this study, the flexible beam is described by the three-node spatial beam element [50] of ANCF. This element has been validated using experimental tests [27]. In this subsection, a parallelized AD scheme is proposed to evaluate the design sensitivities and Jacobian of the elemental elastic force. The design
parameters (independent variables) of the ANCF beam element with circular cross section are assumed to be

$$a = \begin{bmatrix} L & R_{\text{in}} & R_{\text{ou}} & E & v \end{bmatrix},$$  \hfill (28)

where $L$ is the length of beam element, $R_{\text{in}}$ and $R_{\text{ou}}$, respectively, denote the inner and outer radii of the cross section; and $E$ and $v$ in Eq. (28) are the material Young’s modulus and Poisson’s ratio, respectively. The generalized coordinates of the beam element are defined as

$$e = [r^T_{i,\eta} \ r^T_{j,\xi} \ r^T_{j,\eta} \ r^T_{k,\eta} \ r^T_{k,\xi}]^T \in \mathbb{R}^{27},$$  \hfill (29)

where $r_i, r_j$, and $r_k$ are the global position vectors of the nodes, $r_{i,\eta} = \frac{\partial r_i}{\partial \eta}$ and $r_{i,\xi} = \frac{\partial r_i}{\partial \xi}$ are the displacement gradients [50]. Then, the global position vector $r$ of a material point can be expressed as

$$r = [S_1 \ I \ S_2 \ S_3 \ I \ S_4 \ I \ S_5 \ I \ S_6 \ I \ S_7 \ I \ S_8 \ I \ S_9 \ I \ e] = S(\chi)e,$$  \hfill (30)

where $I$ denotes the $3 \times 3$ identity matrix, $\chi = [\xi \ \eta \ \zeta]^T$ denotes the vector defined in the element local coordinate system, $S(\chi) \in \mathbb{R}^{3 \times 27}$ is the element shape function, and

$$S_1 = -\frac{2}{L^2} \left( \frac{L}{2} - \xi \right), \quad S_2 = \eta S_1, \quad S_3 = \zeta S_1,$$
$$S_4 = \frac{2}{L^2} \left( \frac{L}{2} - \zeta \right), \quad S_5 = \eta S_4, \quad S_6 = \zeta S_4,$$
$$S_7 = -\frac{4}{L^2} \left( \frac{L}{2} - \frac{\xi}{2} \left( \frac{L}{2} + \frac{\zeta}{2} \right) \right), \quad S_8 = \eta S_7, \quad S_9 = \zeta S_7.$$  \hfill (31)

To improve the computational efficiency, Eq. (31) can be further rewritten in a compact form as

$$r = \bar{e}^T S(\chi),$$  \hfill (32)

where $\bar{e} \in \mathbb{R}^{9 \times 3}$ reads

$$\bar{e} = \begin{bmatrix} e_1 & e_4 & e_7 & e_{10} & e_{13} & e_{16} & e_{19} & e_{22} & e_{25} \\ e_2 & e_5 & e_8 & e_{11} & e_{14} & e_{17} & e_{20} & e_{23} & e_{26} \\ e_3 & e_6 & e_9 & e_{12} & e_{15} & e_{18} & e_{21} & e_{24} & e_{27} \end{bmatrix}^T,$$  \hfill (33)

and $\bar{S}(\chi) \in \mathbb{R}^{9}$ yields

$$\bar{S}(\chi) = [S_1 \ S_2 \ S_3 \ S_4 \ S_5 \ S_6 \ S_7 \ S_8 \ S_9]^T.$$  \hfill (34)

According to the work by Gerstmayr [51], the deformation gradient of a finite element of ANCF reads

$$G = \frac{\partial r}{\partial r_0} = \frac{\partial r}{\partial \chi} \left( \frac{\partial \chi}{\partial \chi} \right)^{-1} = \left( \bar{e}^T \frac{\partial \bar{S}}{\partial \chi} \right) \left( \frac{\partial \bar{S}}{\partial \chi} \right)^{-1} = \bar{e}^T = H(\chi),$$  \hfill (35)

where $\bar{e}_0$ denotes the vector of initial global coordinates of the nodes, and matrix $H \in \mathbb{R}^{9 \times 3}$ depends on $\chi$ and takes the following form

$$H(\chi) = \frac{\partial \bar{S}}{\partial \chi} \left( \frac{\partial \bar{S}}{\partial \chi} \right)^{-1}.$$  \hfill (36)

Then, according to the continuum mechanics, the components of the vector $F^E \in \mathbb{R}^{27}$ of elemental elastic forces read

$$F^E_i = -\frac{1}{2} \bar{e}_{m} \bar{e}_{j} \bar{e}_{k} \bar{e}_{l} dV \left( D_{abc} H_{ad} H_{be} H_{cf} H_{de} dV \right) + \frac{1}{2} \bar{e}_{m} \int_{V} D_{abc} H_{ad} H_{be} H_{cf} dV,$$  \hfill (37)

where $a, b, c, d, m, n = 1, 2, 3$ and $t, k, f, v = 1, 2, \ldots, 9$, $\delta_{hc}$ is the Kronecker delta function, and the fourth-order isotropic elasticity tensor $D_{abc}$ is defined as

$$D_{abc} = \lambda \delta_{an} \delta_{bc} + \mu (\delta_{ab} \delta_{nc} + \delta_{ac} \delta_{nb}),$$  \hfill (38)

with the Lamé parameters $\lambda = \frac{E_\nu}{(1+\nu)(1-2\nu)}$ and $\mu = \frac{E_\nu}{2(1+\nu)}$. To alleviate the element locking problem, according to the work by Nachbagauer [50, 52], the

Fig. 3 Flowchart of the propagation of variable dependencies for evaluating element elastic forces
elasticity tensor can be firstly split into two parts, and the selective reduced integration technique can be further utilized to evaluate the elemental elastic forces. As shown in Eq. (37), the evaluation of elastic forces can be considered as a volumetric integration of a polynomial function \( f(\xi, \eta, \zeta) \) over the beam element. According to [53], the numerical integration can be calculated by

\[
\int_V f(\xi, \eta, \zeta) \, dV = \frac{\pi LR^2}{n} \sum_{i=1}^{n-1} \sum_{j=1}^{n} \sum_{k=1}^{n} w(i)w(j)w(k)f(\xi(i), \eta(j), \zeta(j,k)).
\]

(39)

Here, the Gauss–Legendre quadrature and Chebyshev–Gauss quadrature are used for the integrals along the beam axis and over the cross section, respectively. Figure 3 shows the flowchart of the propagation of variable dependencies for computing the elemental elastic forces formulated by Eq. (37). The figure only is a brief representation of the “computational graph” mentioned in Sect. 3.3.

Evaluating \( \mathbf{F}^E(\mathbf{e}, \mathbf{a}) \) by Eq. (37) is quite time-consuming because of the complex tensor operations. To obtain the objective sensitivities of SA, the Jacobian of the elastic force \( \partial \mathbf{F}^E / \partial \mathbf{e} \in \mathbb{R}^{27 \times 27} \) and the design sensitivities of the elastic force \( \partial \mathbf{F}^E / \partial \mathbf{a} \in \mathbb{R}^{27 \times 5} \) are both required. Here, \( \partial \mathbf{F}^E / \partial \mathbf{e} \), also known as the tangent stiffness matrix, is nonlinear in general. In this study, a parallelized AD scheme is utilized to evaluate these derivatives. The entries of vectors \( \mathbf{a} \) and \( \mathbf{e} \) are regarded as the independent variables for computing \( \partial \mathbf{F}^E / \partial \mathbf{a} \) and \( \partial \mathbf{F}^E / \partial \mathbf{e} \), respectively. The derivatives can be automatically computed and propagated along the computational graph. Only minor modifications of the original codes used for the computation of vector \( \mathbf{F}^E(\mathbf{e}, \mathbf{a}) \) are required. In Eq. (37), the term \( \int_V \mathbf{D} \mathbf{H}_{\mathbf{a} \mathbf{a}} \mathbf{H}_{\mathbf{e} \mathbf{e}} \mathbf{H}_{\mathbf{e} \mathbf{a}} dV \) is the component of a fourth-order tensor, and all the entries of this tensor can be stored in matrix \( \mathbf{T} \in \mathbb{R}^{81 \times 81} \), which is function of the parameter vector \( \mathbf{a} \). Thus, the invariant derivatives \( \partial \mathbf{T} / \partial \mathbf{a} \) can be computed before the time integration iterations. The invariant derivatives of \( \int_V \mathbf{D} \mathbf{H}_{\mathbf{a} \mathbf{a}} \mathbf{H}_{\mathbf{e} \mathbf{e}} \mathbf{H}_{\mathbf{e} \mathbf{a}} dV \) in Eq. (37) are also evaluated before the time integration iterations likewise.

To improve computational efficiency, the OpenMP directives parallelize the computation tasks of traversing \( \mathbf{T} \in \mathbb{R}^{81 \times 81} \). As shown in Fig. 4, matrix \( \mathbf{T} \) can be divided into several sub-matrices so that the AD of different sub-matrices can be performed in different threads. The independent, intermediate and dependent variables in the computational graph are all extended by using the AD data structure. Thus, the derivatives of each sub-matrix can be computed and propagated in

![Fig. 4 A schematic view of the parallelized automatic differentiation for the matrix T](image-url)
parallel by different threads. $\partial T / \partial a$ can be finally determined from assembling the derivatives of all the sub-matrices.

Figure 5 presents the pseudocode for the computation of $\partial T / \partial a$ using OpenMP. The directive "!$OMP parallel do/end parallel do" specifies which loops the compiler should parallelize. The clause "collapse (2)" specifies how many loops in the nested loops should be collapsed into one large iteration space and parallelized. The clause "private (...)" declares the variables to be private to each thread. The clause "default (shared)" indicates that the unspecified variables are shared among threads.

The above discussions of the Jacobian and design sensitivities are based on the elastic force written in a general tensor form as Eq. (37). The merit of AD approach can also be recognized in computing complex elastic forces $F^E$ of ANCF, which cannot be easily obtained in a closed form. By declaring the nodal coordinates as independent variables and writing the volume integral of strain energy density function explicitly, the complex elastic forces $F^E$ can be obtained effortlessly via the AD approach. The finite elements using high-order interpolations and nonlinear constitutive models [54] can be easily considered by changing the formulae of the element strain energy. Furthermore, the Jacobian $\partial F^E / \partial e$ and design sensitivities $\partial F^E / \partial a$ of the elastic forces can also be conveniently calculated by using the AD approach.

4 Case studies

4.1 The dynamic SA of a spring-actuated five-bar mechanism

This subsection shows a benchmark for the SA of rigid body dynamics [5, 6]. As shown in Fig. 6, the five-bar mechanism is actuated by two linear elastic springs without dampers and is released from the illustrated configuration and subject to vibration under gravitational acceleration, $g = 9.81 \text{m/s}^2$ in $-Y$ direction. The rigid bars are described by using rigid-body elements of ANCF-RN [55]. They have uniformly distributed masses $m_{OA} = m_{CD} = 1 \text{kg}$, $m_{AB} = m_{BC} = 1.5 \text{kg}$, and...
The two springs have identical constant stiffness coefficients $k_1 = k_2 = 100$ N/m. The natural lengths of two springs are chosen to be the design parameters $a = [l_1 \ l_2]^T$, whose reference values are specified as $l_1 = \sqrt{5}$ m and $l_2 = \sqrt{4.25}$ m. According to the works by Dopico et al. [5, 6], the objective function is given as

$$
\psi = \int_0^5 (\mathbf{r}_B - \mathbf{r}_B^0)^T (\mathbf{r}_B - \mathbf{r}_B^0) \, dt,
$$

where $\mathbf{r}_B$ and $\mathbf{r}_B^0$ denote the time-varying and initial global position vectors of point $B$, respectively. The objective function gives the accumulated global position variations of point $B$. Minimizing this objective function is equivalent to design the natural lengths of the two springs so that the initial configuration is the static one. The analytical results of the optimal natural lengths of the two springs can be derived via the principle of virtual work, which are $l_1 = 2.431923574$ and $l_2 = 1.844868887$.

To begin with, the subsection turns to a comparison of the computed generalized state sensitivities between the finite difference method and the direct differentiation method. Figure 7 shows the time history of the acceleration sensitivities of the $X$-direction component of point $B$ with respect to $l_1$ and $l_2$. Three different perturbation steps are chosen for the finite difference method. The comparison does not touch upon the adjoint variable method here because it circumvents the computation of generalized state sensitivities. Figure 7 clearly shows that the accuracy of the finite difference method strongly depends on the perturbation steps. If the perturbation step is too large or too small, the computed sensitivities will deviate from those obtained by the direct differentiation method.

Furthermore, the effect of integration step $h$ on the obtained objective sensitivities is also studied. As shown in Table 2, with different integration step sizes, the objective sensitivities obtained by the direct differentiation method and the adjoint variable method are compared. The convergent objective sensitivities obtained from the two methods are in good agreement with each other, which indicates that the numerical sensitivities obtained by the deducted analytical formulations of SA are correct and reliable.

### 4.2 The static SA of a flexible cantilever beam

To demonstrate the accuracy and efficiency of the proposed AD method in Sect. 3.4, the static SA of a flexible cantilever beam under a small vertical tip load is provided.
As shown in Fig. 8, a slender beam made of a material with Young’s modulus $E = 2 \times 10^{11}$ Pa, subject to a tip load $F = 10$ N, has length $L = 1$ m, but two types of cross section. The rectangular cross section has two design parameters, the height and width $H=W=0.01$ m. The circular cross section has one design parameter $R = 0.01$ m.

If the beam length, cross section and Young’s modulus are taken for parametric design, the vector of design parameters of a rectangular beam can be written as $\mathbf{a} = [L, H, W, E]$, and that of a circular beam can be written as $\mathbf{a} = [L, R, E]$. The beam is meshed by the locking-free beam elements described in Sect. 3.4.

The objective function for static sensitivity analysis is the elastic deflection $w$ of the free end. The closed-form solution of $w$ reads

$$w = -\frac{FL^3}{3EI_z},$$

(41)

where the second moment of area is $I_z = HW^3/12$ for a rectangular cross section and $I_z = \pi R^4/4$ for a circular cross section. Moreover, the deflection sensitivity $\partial w/\partial a_i$ can be computed analytically by differentiating Eq. (41) with respect to the corresponding design parameter, which is used as the reference value for the follow-up comparative study. For the rectangular beam, the analytical deflection sensitivities with respect to $L$, $H$, $W$ and $E$ are $-6 \times 10^{-2}$, 2, 6 and $1 \times 10^{-13}$, respectively. For the circular beam, the analytical deflection sensitivities with respect to $L$, $R$ and $E$ are $-4.912 \times 10^{-2}$, 1.092 $\times 10^2$ and 8.187 $\times 10^{-14}$, respectively.

Figure 9 shows the relative errors of the numerical deflection sensitivities with respect to the analytical reference values. The numerical deflection sensitivities are obtained by using the AD method and FD method with both the rectangular and circular cross sections taken into account. As elaborated in Sect. 3.4, the AD method is used to evaluate the design sensitivities of the elastic forces of the beam element. As different physical parameters may have distinctive magnitudes, the perturbation steps of the FD method are chosen as the relative variations in the design values of parameters. As shown in Fig. 9, the deflection sensitivities computed via the AD method perfectly match the reference values, while those obtained via the FD method depend significantly on
the perturbation steps. Also, the AD method computes the sensitivities with respect to different parameters simultaneously, while the FD method requires that each design parameter is perturbed separately.

Figure 10a indicates that the AD of the vector of elastic forces costs the vast majority of the total computation time. Such a fact does not change with an increase in the element numbers. As shown in Fig. 10b, the computation time decreases as the AD of the Jacobian are parallelized by using the OpenMP directives. However, the computational time of AD still occupies a significant portion of the total computation time. Thus, it can be concluded that computing the complex derivatives efficiently and accurately is paramount for the static SA.

4.3 The dynamic SA of a spatial flexible double pendulum

This subsection presents the dynamic SA of a flexible multibody system undergoing both overall motions and large deformations. As shown in Fig. 11, the spatial double pendulum is initially at rest in the horizontal configuration and then moves under gravitational acceleration, \( g = 9.81 \text{ m/s}^2 \) in \(-Z\) direction. Pendulum OA is connected to the inertial frame through a revolute joint, while pendulum OA and OB are connected by a spherical joint. The two pendulums of rectangular cross sections are modeled as flexible bodies and meshed by the locking-free beam elements described in Sect. 3.4. Table 3 lists the model parameters of the two pendulums.

Prior to the sensitivity analysis of this mechanism, the performance of the locking-free beam element is validated via the dynamic simulation of the spatial double pendulum under gravity action. Figure 12 presents the state variable results of point B obtained by the method in this study and the commercial software (MSC.ADAMS) when different Young’s modulus is used for the pendulum AB. The Newmark integrator is used in both cases with the parameters \( \beta = 0.36 \) and \( \gamma = 0.7 \).

As shown in Fig. 12, the results are perfectly matched when the Young’s modulus of the pendulum
**Fig. 12** Comparison of the state variable results for the point B: the method in this study versus the commercial software.

(a) Position results  
(b) Velocity results  
(c) Acceleration results

**Fig. 13** Comparison of $X$-direction position sensitivities of point B obtained via the direct differentiation method and the finite difference method.

(a) $\partial X_B / \partial H$  
(b) $\partial X_B / \partial \rho$  
(c) $\partial X_B / \partial E$
AB is set as $E = 2 \times 10^{11} \text{ Pa}$. When the Young’s moduli of the pendulum AB are set as $E = 2 \times 10^7 \text{ Pa}$ and $E = 2 \times 10^8 \text{ Pa}$, the corresponding result curves obtained by the method in this study and MSC.A-DAMS are slightly different, the reason is that the beam element used in MSC. ADAMS differs from the locking-free three-node element of ANCF.

For the convenience of further discussion, the vector of design parameters is set as $\mathbf{a} = [H \rho E]^T$, where $H$, $\rho$ and $E$ are the height of cross section, density and Young’s modulus of materials of pendulum AB, respectively. Figure 13 presents the comparison of the time histories of the $X$-direction position sensitivities of point B obtained via the direct differentiation method and the finite difference method. Most curves of the two methods in Fig. 13 exhibit good agreements when the perturbation steps of the finite difference method are properly chosen. However, the two methods still exhibit an obvious difference regarding the position sensitivities with respect to $H$, as shown in Fig. 13a. This difference cannot be eradicated after the exhaustive attempts with other perturbation steps of the FD method. That is, the perfect perturbation step of a certain parameter may not exist for the dynamics of a flexible multibody system undergoing overall motion and large deformation since the system usually exhibits fast evolution of dynamic behaviors.

To compare the adjoint variable method with the direct differentiation method and the finite difference method, an objective function is defined as

$$
\psi = \int_0^{t_F} (X_B)^2 \, dt,
$$

where $t_F$ is the termination time of concern, $X_B$ denotes the global position of point B along $X$-direction. Figure 14 shows the comparison among the time-varying objective sensitivities determined from the three methods. The objective sensitivities obtained by the direct differentiation method and the adjoint variable method are in good agreement with each other. It can also be found from Fig. 14a as $t_F$

---

**Fig. 14** Comparison of objective sensitivities among the direct differentiation method, adjoint variable method and finite difference method
increases, the objective sensitivity with respect to $H$ obtained by the finite difference method will gradually deviate from those obtained by the other two methods. This owes to the accumulative errors of generalized state sensitivities.

4.4 The dynamic SA of a deployable seven-module antenna reflector

This subsection demonstrates that the proposed framework is able to apply to the SA of complex rigid-flexible systems. The case study of concern is a deployable seven-module antenna reflector for a satellite to be designed to meet the stiffness, reliability and accuracy requirements. The most important issue is that such a complicated structure has a great number of parameters in design. The initial design of such structures based on experience is far from optimal one.

In this context, sensitivity analysis is quite useful for evaluating the initial design quantitatively. The case study in this subsection provides a paradigm for the sensitivity analysis of a large-scale engineering structure.

As shown in Fig. 15, the satellite antenna reflector consists of seven modules. Module ① is located at the center, and the other six modules from ② to ⑦ are surrounding module ① in a circular pattern. The deployment of each module is driven by an umbrella-like mechanism on the center rod. Moreover, each module is assembled by six identical deployable ribs radiated from the common center rod. The dimensions and topology of a basic deployable rib are also shown in the bottom of Fig. 15. The center rod, connecting rods and joint accessories are modeled as rigid bodies described by ANCF-RN [55]. The upper, lower, diagonal and vertical beams of each rib are modeled as flexible bodies meshed by the locking-free beam elements of ANCF [50]. The kinematic constraints in each module include fixed joints, revolute joints and cylindrical joints. The connections between the adjacent modules are achieved via the revolute joints. The connecting parts between the modules ③ and ④ are fixed in space, which represents a welding connection.

![Seven-module reflector](image)

**Table 4** Reflector multibody model parameters

| Rigid bodies | Finite elements | Joints | Constraint equations |
|--------------|-----------------|--------|----------------------|
| 483          | 1302            | 849    | 10,014               |

In this context, sensitivity analysis is quite useful for evaluating the initial design quantitatively. The case study in this subsection provides a paradigm for the sensitivity analysis of a large-scale engineering structure.

As shown in Fig. 15, the satellite antenna reflector consists of seven modules. Module ① is located at the center, and the other six modules from ② to ⑦ are surrounding module ① in a circular pattern. The deployment of each module is driven by an umbrella-like mechanism on the center rod. Moreover, each module is assembled by six identical deployable ribs radiated from the common center rod. The dimensions and topology of a basic deployable rib are also shown in the bottom of Fig. 15. The center rod, connecting rods and joint accessories are modeled as rigid bodies described by ANCF-RN [55]. The upper, lower, diagonal and vertical beams of each rib are modeled as flexible bodies meshed by the locking-free beam elements of ANCF [50]. The kinematic constraints in each module include fixed joints, revolute joints and cylindrical joints. The connections between the adjacent modules are achieved via the revolute joints. The connecting parts between the modules ③ and ④ are fixed in space, which represents a welding connection.
between the reflector and its satellite platform. Table 4 shows the reflector multibody model parameters. Figure 16 shows the dynamic deployment configurations of the reflector at different simulation moments.

The SA model here has 672 design parameters in total, including the outer radius of cross section of each beam \( R = 5 \times 10^{-3} \text{ m} \), numbered from 1 to 168, material density of each beam \( \rho = 1800 \text{ kg/m}^3 \), numbered from 169 to 336, Young’s modulus of each beam \( E = 1.6 \times 10^{11} \text{ Pa} \), numbered from 337 to 504) and Poisson’s ratio \( v = 0.3 \), numbered from 505 to 672) of each beam. The antenna reflector deploys itself in the non-gravity environment under the driving forces of the springs distributed in each module. To study the effect of the design parameters on the dynamic responses of the deployment process, the objective function is written as

\[
\psi = \int_0^{t_F} \mathbf{r}_C(t)^T \mathbf{r}_C(t) \, dt,
\]

where \( \mathbf{r}_C(t) \) denotes the position vector from the point of welding connection to a remote crucial point \( C \), as shown in Fig. 15. The length of \( \mathbf{r}_C(t) \) grows as the reflector deploys itself. In this study, the final time \( t_F \) and the time step are set as 10 s and \( 1 \times 10^{-4} \text{ s} \), respectively. As the model has 23,436 generalized coordinates and the simulation is set as 100,000 integration steps, the checkpointing scheme described in Sect. 2.4 with 100 check points is utilized to make sure that the cost memories of the adjoint variable method will not exceed the maximum computer memory.

Figure 17 demonstrates the comparison of the objective sensitivities with respect to the design parameters of the flexible beams determined from the direct differentiation method and adjoint variable method. The objective sensitivities obtained via the two methods show good agreement. According to the obtained objective sensitivities, the influence of the design parameters on the dynamic response of antenna reflector can be sorted and evaluated quantitatively. If the objective sensitivities are checked module by module, it can be found that their values conform to the symmetric distribution of the locations of the seven modules. For example, as shown in Fig. 17a, the obtained objective sensitivities for modules \( \odot \) and \( \oplus \) show a similar distribution, and so do modules \( \ominus \) and \( \ominus \), as well as modules \( \odot \) and \( \ominus \). This is because these pairs of modules are symmetric about the fixed joint between the reflector and the inertial space. It is worth noting that the finite difference method is not utilized here for comparison, because it is not practical to perturb all 672 design parameters one by one for such a complicated model.

Fig. 16 Dynamic deployment configurations of the reflector at different simulation moments
Even though the direct method and adjoint method give close objective sensitivities, their computation efficiency is quite different. As shown in Table 5, the total computation time for the adjoint variable method (201,163 s) is about half of the one for the direct differentiation method (392,727 s). Thus, the adjoint variable method is advantageous over the direct differentiation method, especially for the systems with a large number of design parameters. It can also be found from Table 5 that for the adjoint variable method, although the checkpointing scheme requires the backward integration of the adjoint DAEs, the effort is necessary and worthy since the total computation time is significantly reduced compared with the direct differentiation method.

Fig. 17 Comparison of the objective sensitivities determined from the direct differentiation method and adjoint variable method

| Table 5 Computation time for the dynamic SA of the reflector model |
|---------------------------------------------------------------|
| Computation time (s) | Direct differentiation method | Adjoint variable method |
|----------------------|-------------------------------|------------------------|
| Dynamics             | 62,267                        | 51,830(Forward) + 51,600(Checkpoint) |
| SA                   | 298,796                       | 97,569                 |
| Total                | 392,727                       | 201,163                |
| SA/Total             | 76.1%                         | 48.5%                  |

Even though the direct method and adjoint method give close objective sensitivities, their computation efficiency is quite different. As shown in Table 5, the total computation time for the adjoint variable method (201,163 s) is about half of the one for the direct differentiation method (392,727 s). Thus, the adjoint variable method is advantageous over the direct differentiation method, especially for the systems with a large number of design parameters. It can also be found from Table 5 that for the adjoint variable method, although the checkpointing scheme requires the backward integration of the adjoint DAEs, the effort is necessary and worthy since the total computation time is significantly reduced compared with the direct differentiation method.
5 Concluding remarks

The study presents a computational framework for the sensitivity analysis of deployable flexible space structures with a large number of design parameters. Based on the differential algebraic equations of index-3, the analytical formulations of the direct differentiation method and the adjoint variable method are derived. The accuracy and applicability of these formulations are verified and evaluated via three numerical examples. In the adjoint variable method, the proposed checkpointing scheme successfully alleviates the cost of computer memory. The numerical results show that the accuracy of the finite difference method strongly depends on the perturbation step, thus, it is less reliable than the direct differentiation method and the adjoint variable method. Moreover, a parallelized automatic differentiation algorithm is proposed for efficiently computing the design sensitivities and Jacobian of the nonlinear elastic force. The effectiveness and efficiency of this algorithm are validated by a cantilever beam example with closed-form solutions. Finally, the proposed methods are used to perform the sensitivity analysis of a seven-module deployable satellite antenna reflector with a large number of design parameters. It was found that the adjoint variable method is more efficient than the direct differentiation method. The obtained objective sensitivities can be used to improve the reflector’s deployment dynamics. Some problems are still open for the sensitivity analysis of the deployable flexible space structures, such as the consideration of stress and strain in the sensitivity objective functions, as well as the sensitivity analysis for the contacting bodies.

Acknowledgements This research was supported in part by the National Natural Science Foundations of China under Grants 11722216 and 11832005. The research was also supported in part by the 111 China Project (B16003). This work was supported in part by the project from the Beijing Institute of Technology-Shanghai Academy of Spaceflight Technology joint laboratory.

Availability of data and materials The datasets generated during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Mitsugi, J., Ando, K., Senbokuya, Y., Meguro, A.: Deployment analysis of large space antenna using flexible multibody dynamics simulation. Acta Astronaut. 47, 19–26 (2000)
2. Li, P., Liu, C., Tian, Q., Hu, H., Song, Y.: Dynamics of a deployable mesh reflector of satellite antenna: parallel computation and deployment simulation1. J. Comput. Nonlinear Dyn. 11, 061005 (2016)
3. Peng, Y., Zhao, Z., Zhou, M., He, J., Yang, J., Xiao, Y.: Flexible multibody model and the dynamics of the deployment of mesh antennas. J. Guid. Control Dyn. 40, 1499–1510 (2017)
4. Li, K., Tian, Q., Shi, J., Liu, D.: Assembly dynamics of a large space modular satellite antenna. Mech. Mach. Theory. 142, 103601 (2019)
5. Dopico, D., Sandu, A., Sandu, C., Zhu, Y.T.: Sensitivity analysis of multibody dynamic systems modeled by ODEs and DAEs. In: Proceedings of the ECCOMAS Thematic Conference on Multibody Dynamics. pp. 1–32. Zagreb, Croatia (2014)
6. Dopico, D., González, F., Luaces, A., Saura, M., García-Vallejo, D.: Direct sensitivity analysis of multibody systems with holonomic and nonholonomic constraints via an index-3 augmented Lagrangian formulation with projections. Nonlinear Dyn. 93, 2039–2056 (2018)
7. Dopico, D., Zhu, Y., Sandu, A., Sandu, C.: Direct and adjoint sensitivity analysis of ordinary differential equation multibody formulations. J. Comput. Nonlinear Dyn. 10, 011012 (2015)
8. Bhalerao, K.D., Poursina, M., Anderson, K.S.: An efficient direct differentiation approach for sensitivity analysis of flexible multibody systems. Multibody Syst. Dyn. 23, 121–140 (2010)
9. Neto, M.A., Ambroísio, J.A.C., Leal, R.P.: Sensitivity analysis of flexible multibody systems using composite materials components. Int. J. Numer. Methods Eng. 77, 386–413 (2009)
10. Xiang, W., Yan, S., Wu, J., Niu, W.: Dynamic response and sensitivity analysis for mechanical systems with clearance joints and parameter uncertainties using Chebyshev polynomials method. Mech. Syst. Signal Process. 138, 106596 (2020)
11. Tromme, E., Bruls, O., Emonds-Alt, J., Bruyneel, M., Virlez, G., Duysinx, P.: Discussion on the optimization problem formulation of flexible components in multibody systems. Struct. Multidiscip. Optim. 48, 1189–1206 (2013)
12. Ebrahimi, M., Butscher, A., Cheong, H., Iorio, F.: Design optimization of dynamic flexible multibody systems using the discrete adjoint variable method. Comput. Struct. 213, 82–99 (2019)
13. Zhang, M., Peng, H., Song, N.: Semi-analytical sensitivity analysis approach for fully coupled optimization of flexible multibody systems. Mech. Mach. Theory. 159, 104256 (2021)
14. Shourijeh, M.S.: Optimal Control and Multibody Dynamic Modelling of Human Musculoskeletal Systems. Ph.D. thesis, University of Waterloo (2013)
15. Serban, R., Freeman, J.S.: Identification and identifiability of unknown parameters in multibody dynamic systems. Multibody Syst. Dyn. 5, 335–350 (2001)

16. Ebrahimi, S., Kövecses, J.: Sensitivity analysis for estimation of inertial parameters of multibody mechanical systems. Mech. Syst. Signal Process. 24, 19–28 (2010)

17. Burden, R.L., Faires, J.D.: Numerical Analysis. Brooks/Cole, Cengage Learning, Boston (2011)

18. Schaffer, A.S.: On the adjoint formulation of design sensitivity analysis of multibody dynamics. Ph.D. thesis, University of Iowa (2005)

19. Laflin, J.J., Anderson, K.S., Khan, I.M., Poursina, M.: Advances in the application of the divide-and-conquer algorithm to multibody system dynamics. J. Comput. Nonlinear Dyn. 9, 041003 (2014)

20. Callejo, A., Dopico, D.: Direct sensitivity analysis of multibody systems: a vehicle dynamics benchmark. J. Comput. Nonlinear Dyn. 14, 021004 (2019)

21. Nachbagauer, K., Oberpeilsteiner, S., Steiner, W.: The use of the adjoint method for solving typical optimization problems in multibody dynamics. J. Comput. Nonlinear Dyn. 10, 061011 (2015)

22. Lauß, T., Oberpeilsteiner, S., Steiner, W., Nachbagauer, K.: The discrete adjoint gradient computation for optimization problems in multibody dynamics. J. Comput. Nonlinear Dyn. 12, 031016 (2017)

23. Serban, R., Hindmarsh, A.C.: CVODES: The Sensitivity-Enabled ODE Solver in SUNDIALS. In: Volume 6: 5th International Conference on Multibody Systems, Nonlinear Dynamics, and Control. (1997)

24. Hindmarsh, A.C., Brown, P.N., Grant, K.E., Lee, S.L., Serban, R., Shumaker, D.E., Woodward, C.S.: SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers. ACM Trans. Math. Softw. 31, 363–396 (2005)

25. Sirkes, Z., Tziperman, E.: Finite difference of adjoint or adjoint of finite difference? Mon. Weather Rev. 125, 6 (1997)

26. Pi, T., Zhang, Y., Chen, L.: First order sensitivity analysis of flexible multibody systems using absolute nodal coordinate formulation. Multibody Syst. Dyn. 27, 153–171 (2012)

27. Bauchau, O.A., Han, S., Mikkola, A., Matikainen, M.K., Gruber, P.: Experimental validation of flexible multibody dynamics beam formulations. Multibody Syst. Dyn. 34, 373–389 (2015)

28. Serban, R., Recuero, A.: Sensitivity analysis for hybrid systems and systems with memory. J. Comput. Nonlinear Dyn. 14, 091006 (2019)

29. Corner, S., Sandu, C., Sandu, A.: Adjoint sensitivity analysis of hybrid multibody dynamical systems. Multibody Syst. Dyn. 49, 395–420 (2020)

30. Sonnevile, V., Brüls, O.: Sensitivity analysis for multibody systems formulated on a Lie group. Multibody Syst. Dyn. 31, 47–67 (2014)

31. Lauß, T., Oberpeilsteiner, S., Steiner, W., Nachbagauer, K.: The discrete adjoint method for parameter identification in multibody system dynamics. Multibody Syst. Dyn. 42, 397–410 (2018)

32. Azari Nejat, A., Moghadasi, A., Held, A.: Adjoint sensitivity analysis of flexible multibody systems in differential-algebraic form. Comput. Struct. 228, 106148 (2020)

33. Wang, Q., Yu, W.: Sensitivity Analysis of Geometrically Exact Beam Theory (GEBT) Using the Adjoint Method with Hydra. In: 52nd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference. American Institute of Aeronautics and Astronautics, Denver, Colorado (2011)

34. Gutiérrez-López, M.D., Callejo, A., de Jalón, J.G.: Computation of independent sensitivities using Maggi’s formulation. In: Proceedings of the 2nd Joint International Conference on Multibody System Dynamics, Stuttgart, Germany (2012)

35. Shabana, A.A., Xu, L.: Rotation-based finite elements: reference-configuration geometry and motion description. Acta Mech. Sin. 37, 105–126 (2021)

36. Martins, J.R.R.A., Hwang, J.T.: Review and unification of methods for computing derivatives of multidisciplinary computational models. AIAA J. 51, 2582–2599 (2013)

37. Baydin, A.G., Pearlmutter, B.A., Radul, A.A., Siskind, J.M.: Automatic differentiation in machine learning: a survey. J. Mach. Learn. Res. 18, 5595–5637 (2017)

38. Griewank, A., Walther, A.: Evaluating derivatives: principles and techniques of algorithmic differentiation. Society for Industrial and Applied Mathematics, Philadelphia, PA (2008)

39. Yu, W., Blair, M.: DNAD, a simple tool for automatic differentiation of fortran codes using dual numbers. Comput. Phys. Commun. 184, 1446–1452 (2013)

40. Straka, C.W.: ADP95: Tool for automatic differentiation of a FORTRAN code designed for large numbers of independent variables. Comput. Phys. Commun. 168, 123–139 (2005)

41. Stamatiadis, S., Prosmiti, R., Farantos, S.C.: AUTO_DERIV: tool for automatic differentiation of a FORTRAN code. Comput. Phys. Commun. 127, 343–355 (2000)

42. Bischof, C., Khademi, P., Mauer, A., Carle, A.: Adifor 2.0: automatic differentiation of Fortran 77 programs. IEEE Comput. Sci. Eng. 3, 18–32 (1996)

43. Utke, J., Aachen, T.H., Fagan, M., Tallent, N., Strout, M., Hill, P.H.C., Fagan, M., Tallent, N., Univer, R.: OpenAD/F: a modular, open-source tool for automatic differentiation of Fortran codes. ACM Trans Math Softw. 34, 1–36 (2008)

44. Callejo, A., Narayanan, S.H.K., García de Jalón, J., Norris, B.: Performance of automatic differentiation tools in the dynamic simulation of multibody systems. Adv. Eng. Softw. 73, 35–44 (2014)

45. Kudruss, M., Manns, P., Kirches, C.: Efficient derivative evaluation for rigid-body dynamics based on recursive algorithms subject to kinematic and loop constraints. IEEE Control Syst. Lett. 3, 619–624 (2019)

46. Hoffait, S., Brüls, O., Granville, D., Cugnon, F., Kerschen, G.: Dynamic analysis of the self-locking phenomenon in tape-spring hinges. Acta Astronaut. 66, 1125–1132 (2010)

47. Newmark, N.M.: A method of computation for structural dynamics. J. Eng. Mech. Div. 85, 67–94 (1959)

48. Boopathy, K., Kennedy, G.J.: Parallel finite element framework for rotorcraft multibody dynamics and discrete adjoint sensitivities. AIAA J. 57, 3159–3172 (2019)

49. Gavrea, B., Negrut, D., Potra, F.A.: The Newmark Integration Method for Simulation of Multibody Systems: Analytical Considerations. In: Design Engineering, Parts A
50. Nachbagauer, K., Gruber, P., Gerstmayr, J.: Structural and Continuum mechanics approaches for a 3D shear deformable ANCF beam finite element: application to static and linearized dynamic examples. J. Comput. Nonlinear Dyn. 8, 0004 (2013)

51. Gerstmayr, J., Shabana, A.A.: Efficient integration of the elastic forces and thin three-dimensional beam elements in the absolute nodal coordinate formulation. In: ECCOMAS Thematic Conference, Madrid, Spain, 21–24 June (2005)

52. Tang, Y., Hu, H., Tian, Q.: A condensed algorithm for adaptive component mode synthesis of viscoelastic flexible multibody dynamics. Int. J. Numer. Methods Eng. 122, 609–637 (2021)

53. Lan, P., Tian, Q., Yu, Z.: A new absolute nodal coordinate formulation beam element with multilayer circular cross section. Acta Mech. Sin. 36, 82–96 (2020)

54. Maqueda, L.G., Shabana, A.A.: Poisson modes and general nonlinear constitutive models in the large displacement analysis of beams. Multibody Syst. Dyn. 18, 375–396 (2007)

55. Shabana, A.A.: ANCF reference node for multibody system analysis. Proc. Inst. Mech. Eng. Part K J. Multi-Body Dyn. 229, 109–112 (2015)

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.