



A discrete adjoint gradient approach for equality and inequality constraints in dynamics

Daniel Lichtenecker¹ · Karin Nachbagauer^{2,3}

Received: 2 October 2023 / Accepted: 15 January 2024 / Published online: 29 January 2024
© The Author(s) 2024

Abstract

The optimization of multibody systems requires accurate and efficient methods for sensitivity analysis. The adjoint method is probably the most efficient way to analyze sensitivities, especially for optimization problems with numerous optimization variables. This paper discusses sensitivity analysis for dynamic systems in gradient-based optimization problems. A discrete adjoint gradient approach is presented to compute sensitivities of equality and inequality constraints in dynamic simulations. The constraints are combined with the dynamic system equations, and the sensitivities are computed straightforwardly by solving discrete adjoint algebraic equations. The computation of these discrete adjoint gradients can be easily adapted to deal with different time integrators. This paper demonstrates discrete adjoint gradients for two different time-integration schemes and highlights efficiency and easy applicability. The proposed approach is particularly suitable for problems involving large-scale models or high-dimensional optimization spaces, where the computational effort of computing gradients by finite differences can be enormous. Three examples are investigated to validate the proposed discrete adjoint gradient approach. The sensitivity analysis of an academic example discusses the role of discrete adjoint variables. The energy optimal control problem of a nonlinear spring pendulum is analyzed to discuss the efficiency of the proposed approach. In addition, a flexible multibody system is investigated in a combined optimal control and design optimization problem. The combined optimization provides the best possible mechanical structure regarding an optimal control problem within one optimization.

Keywords Discrete adjoint method · Sensitivity analysis · Optimal control · Optimal design · Flexible multibody systems

✉ D. Lichtenecker
daniel.lichtenecker@tum.de

K. Nachbagauer
karin.nachbagauer@fh-wels.at

¹ TUM School of Engineering and Design, Department of Mechanical Engineering, Chair of Applied Mechanics, Munich Institute of Robotics and Machine Intelligence (MIRMI), Technical University of Munich, Munich, Germany

² Institute for Advanced Study, Technical University of Munich, Lichtenbergstraße 2a, 85748 Garching, Germany

³ Faculty of Engineering and Environmental Sciences, University of Applied Sciences Upper Austria, Stelzhamerstraße 23, 4600 Wels, Austria

1 Introduction

The direct differentiation and the adjoint variable approach represent two principal analytical methods employed for sensitivity analysis in the context of optimization. The direct differentiation method [13] facilitates implementation via straightforward differentiation of system equations, constraints, and cost functions with respect to optimization variables. Despite this, the direct differentiation yields a tremendous computational effort for large-scale optimization problems. Alternatively, the adjoint variable method [13] determines design sensitivities as the solution of adjoint variable equations deduced from variations of the system equations. This avoids the necessity for direct computation of state sensitivities and can dramatically reduce the computational effort required for large-scale optimization problems when adjoint variable equations and algorithms are properly formulated.

The computation of gradients in optimization problems includes the adjoint method with a long history in optimal control theory [24]. Adjoint gradients, e.g., applied for trajectory planning, have already been presented in 1975 by Bryson and Ho [4]. The adjoint gradients have become more and more important in various optimization problems in engineering, e.g., in design optimization [32] or multibody dynamics [7, 10], since the applications have become larger in dimension. Therefore, an efficient calculation of gradients has become essential. Sensitivity analysis has wide-ranging applications in science and engineering, including optimization, parameter identification, data assimilation, optimal control, uncertainty analysis, and experimental design. Current trends in neural networks benefit from solvers capable of building efficient gradient computation for training machine-learning embedded cost functionals in high dimensions [36]. Johnston and Patel stated in [17] that adjoint methods are used both in control theory and machine learning to efficiently compute gradients of functionals. Recent publications discuss the adjoint method in multibody dynamics for various applications, e.g., in a feedback–feedforward control to compute the input control signal and corresponding trajectory predicted by a model [27]. In the work by Schneider and Betsch [38], the choice of a mechanical Hamiltonian and the incorporation of constraints is discussed, and a new approach that preserves the variational structure of the problem is introduced. Moreover, an adjoint sensitivity analysis using a QR decomposition in [16] shows how the adjoint variable method can be applied to multibody systems whose system equations are initially set up in differential–algebraic form but solved in minimal coordinates.

For optimization in complex, large-scale optimization problems, a discrete version of the adjoint method with neat features in terms of stability and accuracy has been proposed recently by various authors [3, 6, 21]. In the discrete adjoint method, the adjoint differential equations are replaced by algebraic equations by introducing a finite-difference scheme for the adjoint system directly from the numerical time-integration method. The method provides exact gradients of the discretized cost function subjected to the discretized equations of motion. The equations of motion of the multibody system and adjoint equations may either be separately discretized from their representations as differential–algebraic equations, or the equations of motion of the multibody system may be discretized first, and the discrete adjoint equations are then derived directly from the discrete multibody equations, tracing back to [4]. It has been emphasized, e.g., by Callejo et al. [6], that the adjoint method is one of the most efficient methods to evaluate sensitivities for problems involving numerous design parameters and relatively few objective functions. The latter paper has presented a discrete version of the adjoint method, which can be applied to the dynamic simulation of flexible multibody systems not only by using an ad hoc backward integration solver but leads to a straightforward algebraic procedure that provides the desired design sensitivities of rigid and flexible multibody systems. Moreover, in [3], the discrete adjoint method

is discussed in different time-marching schemes, including backward difference formulas, Newmark and Adams–Bashforth–Moulton methods. In [25], a discrete adjoint sensitivity analysis considering a Newmark family integrator is presented. In Lauß et al. [21], the discrete adjoint equations for the computation of gradients of a cost function are derived using the Hilbert–Hughes–Taylor (HHT) solver to solve the system equations. The great advantage of this approach is that the cost function can also depend on the accelerations, thus allowing the use of measured data from acceleration sensors in the optimization procedure in a straightforward manner.

Flexible multibody formulations must be included in optimization problems when the cost function includes elasticity or deformation of mechanical systems. When dealing with flexible multibody systems with large deformations or large rotations, the absolute nodal coordinate formulation (ANCF) [40] is advantageous because the formulation does not use rotational degrees of freedom. Using these ANCF elements for flexible bodies, a gradient-based adjoint optimization approach has been presented by Held and Seifried [15]. There, a criterion accounts for the deformation energy of the flexible body. A recent work [44] presents an optimization approach that exploits the adjoint variable method in combination with the flexible natural coordinates formulation for obtaining the sensitivity information. A comprehensive literature review in [12] presents various gradient-based optimization methods, especially in the design optimization of flexible multibody systems. The latter-mentioned review paper discusses the main goals in the design optimization of flexible multibody dynamics and reviews concepts and applications in this field. Over 160 publications in the bibliography give a comprehensive overview of optimization algorithms, types and formulations, and sensitivity analysis. Optimal control and design optimization are discussed in various publications, but there is a gap in the literature on optimization or sensitivity analysis that combines optimal control and design optimization of multibody systems.

This paper significantly enhances optimal control and design optimization problems for flexible multibody systems. Promising results in a preliminary paper by Lichtenecker et al. [22] have shown an efficient optimal control strategy for highly flexible robotic systems based on the adjoint gradient computation method. Furthermore, flexible multibody systems, e.g., soft robots, allow new potential for performing various tasks. Soft robots have not yet fully demonstrated their capabilities, as nature is still clearly superior to them in some areas, particularly evolutionary improved motion and control. Future research should address critical challenges and focus on understanding the fundamental principles that govern the design, modeling, and control of soft robots, as stated in Hawkes et al. [14] and Della Santina et al. [9]. Improving the performance of mechanical systems, like soft robots, requires sophisticated optimization strategies to fulfill the high demands of current and future product requirements. In general, two problem formulations can be considered to describe various optimization applications: structural optimization of mechanical components and/or finding an optimal control for dynamical systems [43]. The focus of this paper is on the combination of both problems. A combined gradient computation for highly efficient optimal control while optimizing the structural components of a mechanical system within the same computation is presented.

To this end, a discrete adjoint gradient computation considering equality and inequality constraints is developed, which will be able to incorporate, e.g., final conditions and/or stress restrictions in design optimizations. With this novel approach, a sensitivity analysis of constraints with respect to optimization variables using discrete adjoints is possible, and a combined optimal control and optimal design of a mechanical system is realized. Three examples will show the application of the proposed discrete adjoint gradient approach: (1)

an academic example of a one-mass oscillator in a sensitivity analysis, (2) an energy optimal control problem of a nonlinear spring pendulum, and (3) a combined optimal control and optimal sizing problem of a flexible two-arm robot using the ANCF for describing large deformations.

2 Problem description

The increasing industrial relevance of high-end solutions that fulfill a wide range of requirements demands the consideration of novel approaches at an early stage of virtual product development. For instance, finding an optimal control of a flexible multibody system under consideration of final constraints is essential to perform a manipulation according to predefined tasks [22]. In addition, innovative lightweight design requires novel approaches in the field of structural optimization [28]. The performance of mechanical systems, either in optimal control problems and/or structural optimization problems, can be increased by optimization.

An optimization problem aims to find a set of optimization variables $\mathbf{z} = \mathbf{z}^* \in \mathbb{R}^z$ to minimize a defined cost function J with respect to constraints. A standard nonlinear programming (NLP) problem can be formulated by

$$\min_{\mathbf{z}} J(\mathbf{z}) \quad (1)$$

s.t.

$$\mathbf{z}_{\min} \leq \mathbf{z} \leq \mathbf{z}_{\max} \quad (2)$$

$$\hat{\mathbf{g}} = \mathbf{0} \quad (3)$$

$$\hat{\mathbf{h}} \leq \mathbf{0}, \quad (4)$$

wherein (2) imposes lower and upper bounds of optimization variables. Equality and inequality constraints are denoted by $\hat{\mathbf{g}} \in \mathbb{R}^p$ and $\hat{\mathbf{h}} \in \mathbb{R}^q$, respectively. To address an optimal control problem with the above NLP formulation, it is necessary to transform the original infinite-dimensional optimization problem into a finite-dimensional one by using a direct transcription method. In general, transcription methods are categorized into shooting methods and collocation methods. A widely used method is the direct single-shooting method, where only the control is parameterized. The reader is referred to [2] for a detailed description of transcription methods. The NLP problem can be treated by well-known algorithms, e.g., the sequential quadratic programming (SQP) approach or the interior point (IP) method [33]. An optimal point \mathbf{z}^* fulfills the Karush–Kuhn–Tucker (KKT) conditions [18, 19], which are necessary first-order optimality conditions for direct optimization methods.

In this paper, the constraints $\hat{\mathbf{g}} = \hat{\mathbf{g}}(\mathbf{x}, \boldsymbol{\xi})$ and $\hat{\mathbf{h}} = \hat{\mathbf{h}}(\mathbf{x}, \boldsymbol{\xi})$ depend on system parameters $\boldsymbol{\xi} \in \mathbb{R}^l$ and on state variables $\mathbf{x} \in \mathbb{R}^n$ due to a time-dependent control $\mathbf{u} \in \mathbb{R}^m$ of a mechanical system. Thus, evaluating the (in)equality constraints requires the evolution of the system response. The dynamics of mechanical systems can be described with ordinary differential equations (ODE)

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, \boldsymbol{\xi}), \quad \mathbf{x}(0) = \mathbf{x}_0. \quad (5)$$

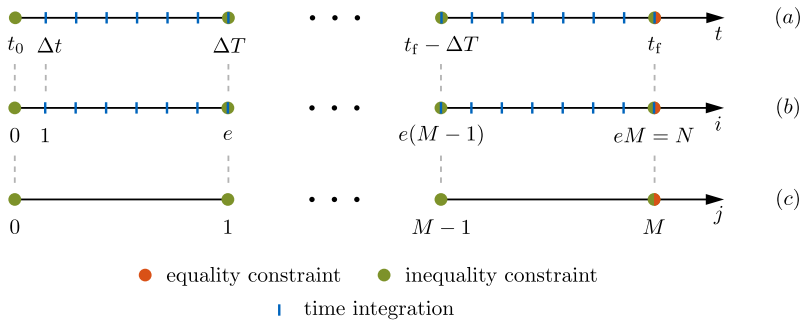


Fig. 1 Time domain and index scale: (a) is the time domain for time integration, (b) is the i -index scale for time integration, and (c) is the j -index scale for the evaluation of constraints

For the numerical computation of the state variables, the ODE can be formulated by a temporal discretization as

$$\mathbf{x}_{i+1} = \tilde{\mathbf{f}}(\mathbf{x}_i, \mathbf{x}_{i+1}, \mathbf{u}_i, \mathbf{u}_{i+1}, \boldsymbol{\xi}), \quad i \in \{0, \dots, N - 1\}, \tag{6}$$

with the given initial state \mathbf{x}_0 . The evolution of the state variables $\mathbf{x}_1, \dots, \mathbf{x}_N$ is influenced by the time integrator used, the control variables $\mathbf{u}_0, \dots, \mathbf{u}_N$, and the set of parameters $\boldsymbol{\xi}$. The discrete ODE (6) represents a general form for an explicit or implicit one-step time integrator, such as a Runge–Kutta scheme [5]. The time domain $t \in [t_0, t_f]$ is discretized with N uniform intervals leading to a constant time-integration step size Δt ; see Fig. 1(a). Consequently, the micro (integration) time mesh is defined by $t_i = \Delta t i, i \in \{0, \dots, N\}$, with $t_0 = 0$ and $t_N = t_f = \Delta t N$.

This paper considers equality constraints at the final time t_f , while inequality constraints must be satisfied on a macro (inequality) time mesh; see Fig. 1(c) for the according index scale of the macro time mesh. The macro time mesh $\hat{t}_j = \Delta T j, j \in \{0, \dots, M\}$ is defined by M time intervals between inequality constraints leading to the inequality step size $\Delta T = t_f/M$. The circumflex ($\hat{\cdot}$) denotes the evaluation of a variable (\cdot) regarding the macro time mesh. The number of chosen time-integration points $e \in \mathbb{N}$ divides the inequality step size ΔT into the integration step size Δt by defining the time-integration step size $\Delta t = \Delta T/e$. For the case $M = N$, one defines inequality constraints at the micro time mesh in each time integration. Figure 1 illustrates the time domain, including the time-integration steps and the arrangement of (in)equality constraints. For example, an equality constraint can be a particular configuration of a mechanical system at the final time t_f . In contrast, an inequality constraint ensures that the acceleration is within a defined limit at the macro time mesh \hat{t} .

For the sake of convenience, the constraints $\hat{\mathbf{g}}$ and $\hat{\mathbf{h}}$ are concatenated into a general set of nonlinear constraints by $\mathbf{c}^T = (\hat{\mathbf{g}}^T, \hat{\mathbf{h}}^T) \in \mathbb{R}^{p+q}$. The equality constraints $\hat{\mathbf{g}} = \mathbf{g}_N$ represent an evaluation of p implicit time-dependent functions $\mathbf{g} = \mathbf{g}(\mathbf{x}(t), \boldsymbol{\xi}) \in \mathbb{R}^p$ at the final time t_f , while the inequality constraints $\hat{\mathbf{h}}$ are a concatenation of r functions $\mathbf{h} = \mathbf{h}(\mathbf{x}(t), \boldsymbol{\xi}) \in \mathbb{R}^r$ evaluated at the macro time mesh. The size of the concatenated inequality constraints is $q = r(M + 1)$; see the j -index scale for inequality constraints in Fig. 1. However, inequality constraints have to be defined in the i -index scale to be in accordance with a time integration of the ODE at the micro time mesh, i.e., inequality constraints are defined in the i -index scale

with $i = ej$. The general set of constraints is formulated in the i -index scale by

$$\mathbf{c} = \mathbf{B}_1 \mathbf{g}_N + \sum_{j=0}^M \mathbf{B}_{2,j} \mathbf{h}_{ej}. \tag{7}$$

Boolean matrices \mathbf{B} map the (in)equality constraints into the combined set of constraints, i.e.,

$$\mathbf{c}^\top = \underbrace{(g_{1,N}, \dots, g_{p,N})}_{\hat{\mathbf{g}}^\top} \underbrace{(h_{1,0}, \dots, h_{r,0}, \dots, h_{1,N}, \dots, h_{r,N})}_{\hat{\mathbf{h}}^\top}, \tag{8}$$

in which the first subscript denotes the row and the second subscript denotes the corresponding time in the i -index scale of \mathbf{g} and \mathbf{h} , respectively.

The constraint formulation in (7) allows a straightforward derivation of discrete adjoint gradients for sensitivity analysis and gradient-based optimizations. In gradient-based optimization algorithms, first-order gradients are crucial to compute a local minimum of an optimization problem. The accuracy of the gradient computation influences the convergence and robustness of optimization algorithms. In addition, the computational effort to solve an optimization problem depends on the runtime required to compute gradients. This paper addresses accurate and efficient first-order sensitivity analysis with particular emphasis on gradients of the constraint formulation in (7). One approach that meets both requirements is the adjoint method [26, 29]. This paper employs a discrete version of the adjoint method to derive first-order gradients of constraints. The proposed discrete adjoint approach replaces the finite-difference approach, usually the default of optimization toolboxes.

3 Sensitivity analysis

This section proposes a novel discrete adjoint approach to sensitivity analysis with emphasis on gradients of the constraint formulation in (7). The sensitivities are computed for a discrete set of optimization variables \mathbf{z} , i.e., the sensitivity analysis is defined by $d\mathbf{c}/d\mathbf{z}$. The adjoint method is an efficient method to compute gradients since the computational effort to compute the so-called adjoint variables does not depend on the number of optimization variables [6, 11, 31]. The discrete adjoint method constructs a finite-difference scheme for the adjoint variables directly from the time-integration method to solve the governing equations [21]. In this paper, the computation of the discrete adjoint gradients is formulated for governing equations in the form of (6). However, the discrete adjoint gradient computation can be easily adapted to deal with different one-step time integrators, as shown for an explicit and an implicit Euler method.

3.1 Discrete adjoint method for one-step time integrators

The goal of the adjoint gradient method is to avoid the expensive computation of state sensitivities $d\mathbf{x}/d\mathbf{z}$ by introducing adjoint variables. Following the fundamental work by Bryson and Ho [4], the constraint (7) is extended by the discrete state equations (6) leading to

$$\tilde{\mathbf{c}} = \mathbf{B}_1 \mathbf{g}_N + \sum_{j=0}^M \mathbf{B}_{2,j} \mathbf{h}_{ej} + \sum_{j=0}^{M-1} \left[\sum_{i=ej}^{e(j+1)-1} \mathbf{R}_{i+1} (\mathbf{x}_{i+1} - \tilde{\mathbf{f}}_{i,i+1}) \right], \tag{9}$$

for any choice of the adjoint variables $\mathbf{R}_1, \dots, \mathbf{R}_N \in \mathbb{R}^{(p+q) \times n}$ in the case where the discrete state equations are satisfied. For the sake of easier reading, the temporal discretized right-hand side of the ODE is defined by $\tilde{\mathbf{f}}_{i,i+1} := \tilde{\mathbf{f}}(\mathbf{x}_i, \mathbf{x}_{i+1}, \mathbf{u}_i, \mathbf{u}_{i+1}, \xi)$, $i \in \{0, \dots, N - 1\}$. The constraint (9) is extended by introducing a double sum to account for the system dynamics. The inner sum considers the time intervals between two inequality constraints $]\hat{t}_j, \hat{t}_{j+1}[$, $j \in \{0, \dots, M - 1\}$ and the outer sum considers exactly the time instances \hat{t}_j , $j \in \{0, \dots, M\}$ at which inequality constraints must be satisfied; see scales in Fig. 1. However, the additional zero terms do not influence the constraints \mathbf{c} in the case where the state equations are satisfied and, therefore, the gradients of \mathbf{c} are equal to the gradients of $\tilde{\mathbf{c}}$. The following derivation of discrete adjoint gradients is based on the extended constraints $\tilde{\mathbf{c}}$.

Before deriving the gradients of $\tilde{\mathbf{c}}$ by a discrete adjoint approach, the boundaries of the state variables \mathbf{x}_N and (in)equality constraints \mathbf{g}_N and \mathbf{h}_N , respectively, must be extracted from (9) to derive terminal conditions for the differential equations of the adjoint variables. Performing an index shift of the inner i -sum in (9) results in an extraction of the boundaries

$$\begin{aligned} \tilde{\mathbf{c}} = & \mathbf{B}_1 \mathbf{g}_N + \mathbf{B}_{2,M} \mathbf{h}_N + \sum_{j=0}^{M-1} \left[\sum_{i=e_{j+1}}^{e^{(j+1)}-1} \mathbf{R}_i \left(\mathbf{x}_i - \tilde{\mathbf{f}}_{i-1,i} \right) \right. \\ & \left. + \mathbf{R}_{e_j} \mathbf{x}_{e_j} - \mathbf{R}_{e^{(j+1)}} \tilde{\mathbf{f}}_{e^{(j+1)}-1,e^{(j+1)}} + \mathbf{B}_{2,j} \mathbf{h}_{e_j} \right] - \mathbf{R}_0 \mathbf{x}_0 + \mathbf{R}_N \mathbf{x}_N. \end{aligned} \tag{10}$$

Note that (9) and (10) are equal despite different formulations. Moreover, note that the subscripts e_j and $e^{(j+1)}$ in (10) are a result of the performed index shift, and the related terms are evaluated in the j -sum at the macro time mesh; see Fig. 1.

The derivation of discrete adjoint gradients is based on the calculus of variations. The first-order variation of (10) in terms of $\delta \mathbf{x}_i$, $\delta \mathbf{x}_{i-1}$, $\delta \mathbf{u}_i$, $\delta \mathbf{u}_{i-1}$, and $\delta \xi$ is given by

$$\begin{aligned} \delta \tilde{\mathbf{c}} = & \mathbf{B}_1 \left(\frac{\partial \mathbf{g}_N}{\partial \mathbf{x}_N} \delta \mathbf{x}_N + \frac{\partial \mathbf{g}_N}{\partial \xi} \delta \xi \right) + \mathbf{B}_{2,M} \left(\frac{\partial \mathbf{h}_N}{\partial \mathbf{x}_N} \delta \mathbf{x}_N + \frac{\partial \mathbf{h}_N}{\partial \xi} \delta \xi \right) \\ & + \sum_{j=0}^{M-1} \left[\sum_{i=e_{j+1}}^{e^{(j+1)}-1} \mathbf{R}_i \left(\delta \mathbf{x}_i - \frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \mathbf{x}_{i-1}} \delta \mathbf{x}_{i-1} - \frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \mathbf{u}_{i-1}} \delta \mathbf{u}_{i-1} - \frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \mathbf{x}_i} \delta \mathbf{x}_i \right. \right. \\ & \left. \left. - \frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \mathbf{u}_i} \delta \mathbf{u}_i - \frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \xi} \delta \xi \right) + \mathbf{R}_{e_j} \delta \mathbf{x}_{e_j} + \mathbf{B}_{2,j} \left(\frac{\partial \mathbf{h}_{e_j}}{\partial \mathbf{x}_{e_j}} \delta \mathbf{x}_{e_j} + \frac{\partial \mathbf{h}_{e_j}}{\partial \xi} \delta \xi \right) \right. \\ & \left. - \mathbf{R}_{e^{(j+1)}} \left(\frac{\partial \tilde{\mathbf{f}}_{e^{(j+1)}-1,e^{(j+1)}}}{\partial \mathbf{x}_{e^{(j+1)}-1}} \delta \mathbf{x}_{e^{(j+1)}-1} + \frac{\partial \tilde{\mathbf{f}}_{e^{(j+1)}-1,e^{(j+1)}}}{\partial \mathbf{u}_{e^{(j+1)}-1}} \delta \mathbf{u}_{e^{(j+1)}-1} \right. \right. \\ & \left. \left. + \frac{\partial \tilde{\mathbf{f}}_{e^{(j+1)}-1,e^{(j+1)}}}{\partial \mathbf{x}_{e^{(j+1)}}} \delta \mathbf{x}_{e^{(j+1)}} + \frac{\partial \tilde{\mathbf{f}}_{e^{(j+1)}-1,e^{(j+1)}}}{\partial \mathbf{u}_{e^{(j+1)}}} \delta \mathbf{u}_{e^{(j+1)}} \right. \right. \\ & \left. \left. + \frac{\partial \tilde{\mathbf{f}}_{e^{(j+1)}-1,e^{(j+1)}}}{\partial \xi} \delta \xi \right) \right] - \mathbf{R}_0 \delta \mathbf{x}_0 + \mathbf{R}_N \delta \mathbf{x}_N. \end{aligned} \tag{11}$$

In this paper, the focus lies on a combined sensitivity analysis with respect to the system parameters ξ and a discrete set of control grid nodes $\bar{\mathbf{u}}$. Following Lichtenecker et al. [22],

the continuous control function is formulated with $\mathbf{u} = \mathbf{C}\bar{\mathbf{u}}$, where \mathbf{C} is a time-dependent interpolation function. The variables of interest in the sensitivity analysis are combined in the vector $\mathbf{z}^T = (\boldsymbol{\xi}^T, \bar{\mathbf{u}}^T)$ and can be used as optimization variables in gradient-based optimization problems. Note that the optimization variables \mathbf{z} can consist of system parameters, e.g., the stiffness of a spring of a mechanical system and/or a parameterization of the control.

To derive a variation of the constraints with respect to the optimization variables \mathbf{z} , the variations $\delta\boldsymbol{\xi}$ and $\delta\mathbf{u}$ in (11) can be obtained in terms of $\delta\mathbf{z}$ with

$$\delta\boldsymbol{\xi} = \mathbf{B}_3\delta\mathbf{z}, \tag{12}$$

$$\delta\mathbf{u} = \mathbf{C}\delta\bar{\mathbf{u}} = \mathbf{C}\mathbf{B}_4\delta\mathbf{z}, \tag{13}$$

respectively. The Boolean matrices \mathbf{B} map the combined set of optimization variables to system parameters and control grid nodes. Substituting (12) and (13) into (11) and reformulating leads to

$$\begin{aligned} \delta\bar{\mathbf{c}} = & \sum_{j=0}^{M-1} \left[\sum_{i=e_j+1}^{e^{(j+1)}-1} \left[\left(\mathbf{R}_i - \mathbf{R}_i \frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \mathbf{x}_i} - \mathbf{R}_{i+1} \frac{\partial \tilde{\mathbf{f}}_{i,i+1}}{\partial \mathbf{x}_i} \right) \delta\mathbf{x}_i \right. \right. \\ & \left. \left. - \mathbf{R}_i \left(\frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \mathbf{u}_{i-1}} \mathbf{C}_{i-1} \mathbf{B}_4 + \frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \mathbf{u}_i} \mathbf{C}_i \mathbf{B}_4 + \frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \boldsymbol{\xi}} \mathbf{B}_3 \right) \delta\mathbf{z} \right] \right. \\ & \left. + \left(\mathbf{R}_{e_j} + \mathbf{B}_{2,j} \frac{\partial \mathbf{h}_{e_j}}{\partial \mathbf{x}_{e_j}} - \mathbf{R}_{e_j+1} \frac{\partial \tilde{\mathbf{f}}_{e_j,e_j+1}}{\partial \mathbf{x}_{e_j}} \right) \delta\mathbf{x}_{e_j} \right. \\ & \left. - \mathbf{R}_{e^{(j+1)}} \left(\frac{\partial \tilde{\mathbf{f}}_{e^{(j+1)}-1,e^{(j+1)}}}{\partial \mathbf{u}_{e^{(j+1)}-1}} \mathbf{C}_{e^{(j+1)}-1} \mathbf{B}_4 + \frac{\partial \tilde{\mathbf{f}}_{e^{(j+1)}-1,e^{(j+1)}}}{\partial \mathbf{u}_{e^{(j+1)}}} \mathbf{C}_{e^{(j+1)}} \mathbf{B}_4 \right. \right. \\ & \left. \left. + \frac{\partial \tilde{\mathbf{f}}_{e^{(j+1)}-1,e^{(j+1)}}}{\partial \boldsymbol{\xi}} \mathbf{B}_3 \right) \delta\mathbf{z} + \mathbf{B}_{2,j} \frac{\partial \mathbf{h}_{e_j}}{\partial \boldsymbol{\xi}} \mathbf{B}_3 \delta\mathbf{z} \right] \\ & + \left(\mathbf{B}_1 \frac{\partial \mathbf{g}_N}{\partial \boldsymbol{\xi}} + \mathbf{B}_{2,M} \frac{\partial \mathbf{h}_N}{\partial \boldsymbol{\xi}} \right) \mathbf{B}_3 \delta\mathbf{z} + \left(\mathbf{B}_1 \frac{\partial \mathbf{g}_N}{\partial \mathbf{x}_N} + \mathbf{B}_{2,M} \frac{\partial \mathbf{h}_N}{\partial \mathbf{x}_N} + \mathbf{R}_N \right) \delta\mathbf{x}_N. \end{aligned} \tag{14}$$

Equation (14) implies the relation between $\delta\mathbf{x}$ and $\delta\mathbf{z}$. The optimization variables \mathbf{z} influence the state variables \mathbf{x} and, therefore, the variation of state variables should be interpreted as [6, 23]

$$\delta\mathbf{x} = \frac{d\mathbf{x}}{d\mathbf{z}} \delta\mathbf{z}. \tag{15}$$

The total derivatives of state variables with respect to optimization variables are obtained by solving the matrix differential equations

$$\frac{d\dot{\mathbf{x}}}{d\mathbf{z}} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \frac{d\mathbf{x}}{d\mathbf{z}} + \frac{\partial \mathbf{f}}{\partial \mathbf{u}} \frac{\partial \mathbf{u}}{\partial \mathbf{z}} + \frac{\partial \mathbf{f}}{\partial \boldsymbol{\xi}} \frac{\partial \boldsymbol{\xi}}{\partial \mathbf{z}}. \tag{16}$$

This system is defined by taking the total derivative of (5) with respect to the optimization variables and, therefore, the dimension of the system depends on the number of optimization

variables. The solution of the matrix differential equations is obtained by applying a temporal discretization, where the computational effort to solve (16) can become expensive in the case of a large number of optimization variables. Using the state sensitivities (16) with (15) in (14), first-order gradients of the constraint formulation in (7) can be obtained without the need for the adjoint variables. This approach is called direct differentiation. However, the goal of the proposed discrete adjoint gradient approach is to avoid the direct computation of the state sensitivities $\frac{d\mathbf{x}}{dz}$. To this end, the discrete adjoint variables in (14) are defined such that the brackets multiplied with $\delta\mathbf{x}$ are zero. Hence, the discrete adjoint variables are obtained by the matrix differential equations

$$\mathbf{R}_N = -\mathbf{B}_1 \frac{\partial \mathbf{g}_N}{\partial \mathbf{x}_N} - \mathbf{B}_{2,M} \frac{\partial \mathbf{h}_N}{\partial \mathbf{x}_N}, \tag{17}$$

$$\mathbf{R}_i = \mathbf{R}_{i+1} \frac{\partial \tilde{\mathbf{f}}_{i,i+1}}{\partial \mathbf{x}_i} + \mathbf{R}_i \frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \mathbf{x}_i}, \quad \forall i \in \{ej + 1, \dots, e(j + 1) - 1\}, \tag{18}$$

$$\mathbf{R}_{ej} = -\mathbf{B}_{2,j} \frac{\partial \mathbf{h}_{ej}}{\partial \mathbf{x}_{ej}} + \mathbf{R}_{ej+1} \frac{\partial \tilde{\mathbf{f}}_{ej,ej+1}}{\partial \mathbf{x}_{ej}}, \quad \forall j \in \{1, \dots, M - 1\}. \tag{19}$$

Equation (17) imposes the terminal condition of the adjoint variables at the final time t_N . The computation of the adjoint variables is performed in a backward manner starting from \mathbf{R}_N and proceeding with the adjoint system (18). It has to be emphasized that (18) is defined within the macro time mesh $[\hat{t}_j, \hat{t}_{j+1}[$, $j \in \{0, \dots, M - 1\}$. The discrete adjoint variables at the macro time mesh \hat{t}_j , $j \in \{1, \dots, M - 1\}$ are determined by the intermediate condition in (19). The adjoint system and the intermediate condition are applied alternately in the backward integration to solve the discrete adjoint variables.

Once the discrete adjoint variables are computed by (17)–(19), the terms related to $\delta\mathbf{x}$ vanish in (14) and, therefore, the variation of the extended constraints simplifies to

$$\delta\bar{\mathbf{c}} = \underbrace{\{\dots\}}_{\frac{d\bar{\mathbf{c}}}{dz}} \delta\mathbf{z}, \tag{20}$$

where the variation $\delta\mathbf{z}$ is factored out. The simplified variation leads to first-order gradients of constraints with respect to optimization variables given by

$$\begin{aligned} \frac{d\bar{\mathbf{c}}}{dz} = & \sum_{j=0}^{M-1} \left[\sum_{i=ej+1}^{e(j+1)-1} \mathbf{R}_i \left(-\frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \mathbf{u}_{i-1}} \mathbf{C}_{i-1} \mathbf{B}_4 - \frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \mathbf{u}_i} \mathbf{C}_i \mathbf{B}_4 - \frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \xi} \mathbf{B}_3 \right) \right. \\ & + \mathbf{B}_{2,j} \frac{\partial \mathbf{h}_{ej}}{\partial \xi} \mathbf{B}_3 - \mathbf{R}_{e(j+1)} \left(\frac{\partial \tilde{\mathbf{f}}_{e(j+1)-1,e(j+1)}}{\partial \mathbf{u}_{e(j+1)-1}} \mathbf{C}_{e(j+1)-1} \mathbf{B}_4 \right. \\ & \left. \left. + \frac{\partial \tilde{\mathbf{f}}_{e(j+1)-1,e(j+1)}}{\partial \mathbf{u}_{e(j+1)}} \mathbf{C}_{e(j+1)} \mathbf{B}_4 + \frac{\partial \tilde{\mathbf{f}}_{e(j+1)-1,e(j+1)}}{\partial \xi} \mathbf{B}_3 \right) \right] \\ & + \left(\mathbf{B}_1 \frac{\partial \mathbf{g}_N}{\partial \xi} + \mathbf{B}_{2,M} \frac{\partial \mathbf{h}_N}{\partial \xi} \right) \mathbf{B}_3. \end{aligned} \tag{21}$$

The discrete adjoint gradient computation for the constraint formulation in (7) is obtained by (21). Note that the computation of discrete adjoint gradients is based on the solution of the adjoint system, whose size does not depend on the number of optimization variables.

The gradient computation using direct differentiation requires the solution of (16), which depends on the number of optimization variables. Therefore, the adjoint-based sensitivity analysis is computationally efficient, especially when dealing with optimization problems with a large number of optimization variables.

Employing the discrete adjoint gradient (21), e.g., within an optimization procedure, requires the specific formulation of the right-hand side vector $\tilde{\mathbf{f}}$. The chosen time integrator to solve the forward dynamics implies the backward integration of the discrete adjoint variables. Moreover, sensitivities of the forward time integrator are recognized in the computation of the discrete adjoint variables. To use the proposed discrete gradient approach, the sensitivities of the chosen forward time integrator need to be defined.

3.2 Application to the explicit Euler method

The explicit Euler method is an iterative solution scheme to approximate the state variables of the ODE in (5) by

$$\mathbf{x}_{i+1} = \underbrace{\mathbf{x}_i + \Delta t \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i, \boldsymbol{\xi})}_{\tilde{\mathbf{f}}(\mathbf{x}_i, \mathbf{u}_i, \boldsymbol{\xi})}. \tag{22}$$

Using the above time-integration scheme in the discrete adjoint sensitivity analysis, the explicit right-hand side vector $\tilde{\mathbf{f}}$ has to be defined in the general form $\tilde{\mathbf{f}}_{i,i+1} := \tilde{\mathbf{f}}(\mathbf{x}_i, \mathbf{u}_i, \boldsymbol{\xi})$. The computation of the discrete adjoint variables requires the derivatives of $\tilde{\mathbf{f}}$ with respect to the state variables \mathbf{x} given by

$$\frac{\partial \tilde{\mathbf{f}}_{i,i+1}}{\partial \mathbf{x}_i} = \mathbf{I} + \Delta t \frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_i} \quad \text{and} \quad \frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \mathbf{x}_i} = \mathbf{0}, \tag{23}$$

where $\mathbf{f}_i = \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i, \boldsymbol{\xi})$ and \mathbf{I} denotes the identity matrix. In addition, the discrete adjoint gradient computation requires the derivatives of $\tilde{\mathbf{f}}$ with respect to \mathbf{u} and $\boldsymbol{\xi}$ given by

$$\frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \mathbf{u}_i} = \mathbf{0}, \quad \frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \mathbf{u}_{i-1}} = \Delta t \frac{\partial \mathbf{f}_{i-1}}{\partial \mathbf{u}_{i-1}} \quad \text{and} \quad \frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \boldsymbol{\xi}} = \Delta t \frac{\partial \mathbf{f}_{i-1}}{\partial \boldsymbol{\xi}}, \tag{24}$$

respectively.

3.3 Application to the implicit Euler method

Similar to the explicit Euler method, the implicit Euler method is an iterative solution scheme to approximate the state variables of the ODE in (5) by

$$\mathbf{x}_{i+1} = \underbrace{\mathbf{x}_i + \Delta t \mathbf{f}(\mathbf{x}_{i+1}, \mathbf{u}_{i+1}, \boldsymbol{\xi})}_{\tilde{\mathbf{f}}(\mathbf{x}_i, \mathbf{x}_{i+1}, \mathbf{u}_{i+1}, \boldsymbol{\xi})}. \tag{25}$$

Using the above time-integration scheme in the discrete adjoint sensitivity analysis, the implicit right-hand side vector $\tilde{\mathbf{f}}$ has to be defined in the general form $\tilde{\mathbf{f}}_{i,i+1} := \tilde{\mathbf{f}}(\mathbf{x}_i, \mathbf{x}_{i+1}, \mathbf{u}_{i+1}, \boldsymbol{\xi})$. The computation of the discrete adjoint variables requires the derivatives of $\tilde{\mathbf{f}}$ with respect to the state variables \mathbf{x} given by

$$\frac{\partial \tilde{\mathbf{f}}_{i,i+1}}{\partial \mathbf{x}_i} = \mathbf{I} \quad \text{and} \quad \frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \mathbf{x}_i} = \Delta t \frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_i}. \tag{26}$$

In addition, the discrete adjoint gradient computation requires the derivatives of $\tilde{\mathbf{f}}$ with respect to \mathbf{u} and ξ given by

$$\frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \mathbf{u}_i} = \Delta t \frac{\partial \mathbf{f}_i}{\partial \mathbf{u}_i}, \quad \frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \mathbf{u}_{i-1}} = \mathbf{0} \quad \text{and} \quad \frac{\partial \tilde{\mathbf{f}}_{i-1,i}}{\partial \xi} = \Delta t \frac{\partial \mathbf{f}_i}{\partial \xi}, \tag{27}$$

respectively.

3.4 Procedure for the use of the discrete adjoint gradients

This section summarizes the sensitivity analysis using the proposed discrete adjoint gradient approach within an optimization problem. The focus is to compute first-order gradients of the constraint formulation in (7) with respect to optimization variables, i.e.,

$$\left. \frac{d\mathbf{c}}{d\mathbf{z}} \right|_{\mathbf{z}=\mathbf{z}^{(k)'}}$$

evaluated at the (k) th iteration in a gradient-based optimization. The use of the proposed discrete adjoint gradient approach can be summarized by the following steps:

1. Set up an optimization problem in the form of (1)–(4) and select an NLP software package to solve the optimization problem, e.g., IPOPT [45].
2. Compute the derivatives by symbolic differentiation for the discrete adjoint approach:
 - a) Compute the derivatives of (in)equality constraints \mathbf{g} and \mathbf{h} with respect to \mathbf{x} and ξ , respectively, by symbolic differentiation.
 - b) Select a numerical time-integration solver and compute the derivatives of \mathbf{f} with respect to \mathbf{x} , \mathbf{u} , and ξ by symbolic differentiation. Additionally, define the derivatives of the solver-specific right-hand side vector $\tilde{\mathbf{f}}$, e.g., as shown in Sect. 3.2 and Sect. 3.3 for the explicit and the implicit Euler method, respectively.
3. Compute the first-order gradients of the constraints using the discrete adjoint approach:
 - a) Compute the state variables \mathbf{x} influenced by the set of optimization variables $\mathbf{z}^{(k)}$ with the chosen time-integration scheme.
 - b) Compute the discrete adjoint variables by solving the matrix differential equations (17)–(19) backward in time.
 - c) Compute the sensitivities of the constraint formulation in (7) using the discrete adjoint gradient approach (21).
4. Provide the cost function, constraints, and the respective first-order gradients via an interface to the chosen NLP software package. The Hessian of the cost function and the constraints are usually computed internally by the software package.
5. Repeat Steps 3 and 4 until the KKT conditions are satisfied and an optimal solution \mathbf{z}^* is found.

As aforementioned in the second step, derivatives with respect to \mathbf{x} , \mathbf{u} , and ξ are computed by symbolic differentiation. Considering a complicated mechanical system with many state variables, the governing equations become extensive and difficult to solve. For such systems, the effort to derive system derivatives by symbolic differentiation is enormous or not feasible in a reasonable time. The following section discusses the symbolic differentiation when a mechanical system is formulated with flexible bodies.

4 Flexible multibody formulation

The governing equations of multibody systems with rigid and flexible bodies are described by the second-order differential equations

$$\mathbf{M}(\mathbf{q}, \xi)\ddot{\mathbf{q}} = \mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{u}, \xi), \tag{28}$$

where \mathbf{M} is the mass matrix, \mathbf{q} denotes the generalized coordinates and \mathbf{Q} is the generalized force vector. In this paper, the generalized force vector

$$\mathbf{Q} = \mathbf{Q}_u + \mathbf{Q}_d + \mathbf{Q}_g - \mathbf{Q}_k \tag{29}$$

consists of the term associated with the control \mathbf{Q}_u , the viscous damping for joint friction \mathbf{Q}_d , the gravity \mathbf{Q}_g , and the elasticity \mathbf{Q}_k . The second-order differential equations are transformed into

$$\dot{\mathbf{x}} = \mathbf{f} = \underbrace{\begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}^{-1} \end{pmatrix}}_{\mathcal{M}^{-1}} \begin{pmatrix} \mathbf{v} \\ \mathbf{Q} \end{pmatrix}, \tag{30}$$

wherein the state variables are expressed by $\mathbf{x}^T = (\mathbf{q}^T, \mathbf{v}^T) \in \mathbb{R}^n$ with the generalized velocities $\dot{\mathbf{q}} = \mathbf{v}$. In this paper, effects due to the elasticity of flexible bodies are considered by a nonlinear finite-element-based formulation using the ANCF as proposed by Omar and Shabana [34]. This standard ANCF element has been tested extensively in the literature and employed in structural-optimization problems, e.g., [15, 41]. The ANCF was developed to solve large-deformation problems in multibody dynamics [40]. Since the ANCF does not use rotational degrees of freedom, the formulation does not necessarily suffer from singularities arising from angular parameterizations. An essential advantage of the ANCF is that the mass matrix is constant with respect to the generalized coordinates, i.e., $\mathbf{M} = \mathbf{M}(\xi)$. For a detailed description of the flexible multibody formulation, the reader is referred to [34].

Employing the proposed discrete adjoint approach to study constraint sensitivities requires first-order derivatives of the governing equations with respect to the states, the control, and the set of parameters; see the procedure provided in Sect. 3.4. In this paper, the system derivatives are computed by using symbolic differentiation for efficient computation in the sensitivity analysis.

Modeling a mechanical system with a large number of structural elements results in an extensive system of governing equations. Therefore, the effort to derive the system derivatives by symbolic differentiation is enormous or not feasible in a reasonable time. Instead of directly computing the derivatives of the governing equations in (30), one can derive the global system derivatives based on the symbolic differentiation of a single structural ANCF element. The element-based derivatives are then assembled to compute the global system derivatives. Therefore, the computation of symbolic differentiations is very efficient and independent of the number of structural elements. To this end, the global mass matrix is defined by the local mass matrix of an element with superscript (e)

$$\mathbf{M} = \sum_{(e)} \mathbf{T}_1^{(e)T} \mathbf{M}^{(e)} \mathbf{T}_1^{(e)}, \tag{31}$$

where the element specific Boolean transformation matrix $\mathbf{T}_1^{(e)}$ maps a global variable to its local representation. Similarly, the global generalized force vector reads

$$\mathbf{Q} = \sum_{(e)} \mathbf{T}_1^{(e)\top} \mathbf{Q}^{(e)}, \tag{32}$$

and the local state variables are given by

$$\mathbf{x}^{(e)} = \begin{pmatrix} \mathbf{q}^{(e)} \\ \mathbf{v}^{(e)} \end{pmatrix} = \begin{pmatrix} \mathbf{T}_1^{(e)} & \mathbf{0} \\ \mathbf{0} & \mathbf{T}_1^{(e)} \end{pmatrix} \begin{pmatrix} \mathbf{q} \\ \mathbf{v} \end{pmatrix} = \tilde{\mathbf{T}}_1^{(e)} \mathbf{x}. \tag{33}$$

In addition, the local control and the local set of parameters are defined by

$$\mathbf{u}^{(e)} = \mathbf{T}_2^{(e)} \mathbf{u} \quad \text{and} \quad \boldsymbol{\xi}^{(e)} = \mathbf{T}_3^{(e)} \boldsymbol{\xi}, \tag{34}$$

respectively, with Boolean transformation matrices. The element-based formulations in (31)–(34) are used in the proceeding section to derive global first-order derivatives based on local representations.

4.1 Element-based derivatives for an efficient implementation of the proposed approach

The computation of the discrete adjoint equations in (17)–(19) and the discrete adjoint gradient in (21) requires the system derivatives with respect to the state variables $\partial \mathbf{f} / \partial \mathbf{x}$, the control $\partial \mathbf{f} / \partial \mathbf{u}$, and the set of parameters $\partial \mathbf{f} / \partial \boldsymbol{\xi}$. As aforementioned, an essential advantage of the ANCF is that the mass matrix does not depend on the generalized coordinates, i.e., the derivatives with respect to the state variables vanish. The constant mass matrix leads to a simplification of the derivatives of the first-order state equations (30) with respect to the state variables by

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \mathcal{M}^{-1} \begin{pmatrix} \mathbf{0} & \mathbf{I} \\ \frac{\partial \mathbf{Q}}{\partial \mathbf{q}} & \frac{\partial \mathbf{Q}}{\partial \mathbf{v}} \end{pmatrix}. \tag{35}$$

To derive element-based derivatives, the derivatives of the global generalized force vector can be formulated with (32) as follows:

$$\frac{\partial \mathbf{Q}}{\partial \mathbf{x}} = \sum_{(e)} \mathbf{T}_1^{(e)\top} \frac{\partial \mathbf{Q}^{(e)}}{\partial \mathbf{x}^{(e)}} \underbrace{\frac{\partial \mathbf{x}^{(e)}}{\partial \mathbf{x}}}_{\tilde{\mathbf{T}}_1^{(e)}}. \tag{36}$$

The symbolic differentiation of the global generalized force vector \mathbf{Q} for all elements simplifies to the symbolic differentiation of the local generalized force vector $\mathbf{Q}^{(e)}$ for one element. The derivatives of the global generalized force vector are assembled by using the element-specific Boolean transformation matrices $\mathbf{T}_1^{(e)}$ and $\tilde{\mathbf{T}}_1^{(e)}$. Thus, the computational effort for symbolic differentiations is tremendously reduced and independent of the number of elements.

In addition, the derivatives of the first-order state equations (30) with respect to the control are given by

$$\frac{\partial \mathbf{f}}{\partial \mathbf{u}} = \mathcal{M}^{-1} \begin{pmatrix} \mathbf{0} \\ \frac{\partial \mathbf{Q}}{\partial \mathbf{u}} \end{pmatrix}, \tag{37}$$

where the derivatives of the generalized force vector with respect to the control are formulated with

$$\frac{\partial \mathbf{Q}}{\partial \mathbf{u}} = \sum_{(e)} \mathbf{T}_1^{(e)\top} \frac{\partial \mathbf{Q}^{(e)}}{\partial \mathbf{u}^{(e)}} \underbrace{\frac{\partial \mathbf{u}^{(e)}}{\partial \mathbf{u}}}_{\mathbf{T}_2^{(e)}}. \tag{38}$$

Finally, the derivatives of the first-order state equations (30) with respect to the set of parameters read

$$\frac{\partial \mathbf{f}}{\partial \boldsymbol{\xi}} = \begin{pmatrix} \mathbf{0} \\ \frac{\partial (\mathbf{M}^{-1} \mathbf{Q})}{\partial \boldsymbol{\xi}} \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \frac{\partial \mathbf{a}}{\partial \boldsymbol{\xi}} \end{pmatrix}, \tag{39}$$

with the generalized accelerations $\mathbf{a} = \mathbf{M}^{-1} \mathbf{Q}$. The derivatives of the generalized accelerations with respect to the set of parameters are difficult to compute because the mass matrix and the generalized force vector are functions of the parameters. The element-based formulation of the generalized accelerations is given by

$$\mathbf{M} \mathbf{a} = \mathbf{Q} \Rightarrow \sum_{(e)} \mathbf{T}_1^{(e)\top} \mathbf{M}^{(e)} \underbrace{\mathbf{T}_1^{(e)} \mathbf{a}}_{\mathbf{a}^{(e)}} = \sum_{(e)} \mathbf{T}_1^{(e)\top} \mathbf{Q}^{(e)}. \tag{40}$$

The matrix–vector product $\mathbf{M}^{(e)} \mathbf{a}^{(e)}$ is reformulated to avoid the direct derivative of the mass matrix with respect to the parameters as a sum of vector–scalar products, leading to

$$\sum_{(e)} \mathbf{T}_1^{(e)\top} \sum_c \mathbf{M}_c^{(e)} a_c^{(e)} = \sum_{(e)} \mathbf{T}_1^{(e)\top} \mathbf{Q}^{(e)}, \tag{41}$$

where the subscript c denotes the column of the mass matrix and the row of generalized accelerations, respectively. The derivatives of the element-based formulation with respect to $\boldsymbol{\xi}$ read

$$\sum_{(e)} \mathbf{T}_1^{(e)\top} \sum_c \left(\frac{\partial \mathbf{M}_c^{(e)}}{\partial \boldsymbol{\xi}^{(e)}} \underbrace{\frac{\partial \boldsymbol{\xi}^{(e)}}{\partial \boldsymbol{\xi}}}_{\mathbf{T}_3^{(e)}} a_c^{(e)} + \mathbf{M}_c^{(e)} \frac{\partial a_c^{(e)}}{\partial \boldsymbol{\xi}} \right) = \sum_{(e)} \mathbf{T}_1^{(e)\top} \frac{\partial \mathbf{Q}^{(e)}}{\partial \boldsymbol{\xi}^{(e)}} \mathbf{T}_3^{(e)}. \tag{42}$$

By reformulation of the dyadic product of an element

$$\sum_c \mathbf{M}_c^{(e)} \frac{\partial a_c^{(e)}}{\partial \boldsymbol{\xi}} = \mathbf{M}^{(e)} \frac{\partial \mathbf{a}^{(e)}}{\partial \boldsymbol{\xi}} = \mathbf{M}^{(e)} \mathbf{T}_1^{(e)} \frac{\partial \mathbf{a}}{\partial \boldsymbol{\xi}}, \tag{43}$$

the derivatives of the global acceleration are computed by substituting (43) into (42) and using (31)

$$\frac{\partial \mathbf{a}}{\partial \boldsymbol{\xi}} = \mathbf{M}^{-1} \sum_{(e)} \mathbf{T}_1^{(e)\top} \left(\frac{\partial \mathbf{Q}^{(e)}}{\partial \boldsymbol{\xi}^{(e)}} \mathbf{T}_3^{(e)} - \sum_c \frac{\partial \mathbf{M}_c^{(e)}}{\partial \boldsymbol{\xi}^{(e)}} \mathbf{T}_3^{(e)} a_c^{(e)} \right). \tag{44}$$

Note that the derivatives are time-dependent functions, but the inverse of the global mass matrix only needs to be computed once for the numerical evaluation since the mass matrix is constant for the ANCF.

The element-based derivatives in (36), (38), and (44) are computed by symbolic differentiation and used to assemble the global derivatives required for the proposed discrete adjoint gradient computation. In this paper, the formulation of the flexible multibody and all required symbolic differentiations are written in the computer algebra system toolbox SymPy [30], which is written in pure Python. In addition, the analytical expressions are compiled ahead of (simulation) time with Numba [20] to efficiently evaluate the derived terms. Both packages are available under an open-source license.

5 Numerical examples

This section discusses three examples to demonstrate the use and advantages of the proposed discrete adjoint gradient approach. In the first example, the sensitivity analysis of an academic one-mass oscillator is analyzed. This example provides a deep insight into the proposed approach and discusses the role of the discrete adjoint variables. The second example studies the energy optimal control problem of a nonlinear spring pendulum to discuss the efficiency of the proposed approach. The third example analyzes a combined optimal control and design problem of a Selective Compliance Assembly Robot Arm (SCARA) in a rest-to-rest motion. The bodies of the robot are modeled with flexible components, where the ANCF discussed in Sect. 4 is used to examine the effects due to elasticity. Explicit and implicit integration schemes are applied to demonstrate the versatility of the proposed approach to different time integrators, with a particular focus on the computation of adjoint variables. In addition, all examples are used to verify the proposed discrete adjoint gradients using numerical computed gradients via the finite-difference method.

5.1 Sensitivity analysis of a one-mass oscillator

As a first example, the proposed discrete adjoint gradient approach in Sect. 3 is applied to the sensitivity analysis of an academic one-mass oscillator to demonstrate the proposed procedure in Sect. 3.4 and to discuss the role of the discrete adjoint variables. The mechanical system consists of a mass m , a linear damping parameter d , and a linear spring parameter c . The mass is driven by a time-dependent control u . The state equations are given by a linear first-order differential system

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, u) = \begin{pmatrix} v \\ \frac{1}{m}(u - dv - cx) \end{pmatrix}, \quad (45)$$

wherein the state variables are expressed by $\mathbf{x} = (x, v)^\top$ with the position x and velocity v of the mass. The control variable is formulated as proposed in [22] with $u = \mathbf{C}\hat{\mathbf{u}}$, where \mathbf{C} is a time-dependent cubic spline interpolation function and $\hat{\mathbf{u}} = (\hat{u}_0, \dots, \hat{u}_M)^\top$ is a set of control grid nodes. The grid nodes are defined at the uniformly distributed macro time mesh $\hat{t}_j, j \in \{0, \dots, M\}$ within the time interval $[t_0, t_f]$.

This example examines sensitivities of the inequality constraint $h = f - f_{\max} \leq 0$ regarding the reaction force $f = c x + d v$ ($r = 1$ time-dependent function) with respect to the grid nodes $\hat{\mathbf{u}}$. Hence, the variables of interest in the sensitivity analysis are defined by $\mathbf{z} = \hat{\mathbf{u}}$. The reaction force is evaluated at the macro time mesh \hat{t}_j , resulting in the concatenated vector $\hat{\mathbf{h}} = (\hat{h}_0, \dots, \hat{h}_M)^\top$. The reaction force depends on the state variables, which are time integrated under the influence of the control. Therefore, a change in the control grid nodes leads to a change in the reaction force. A graphical illustration of the dependencies is

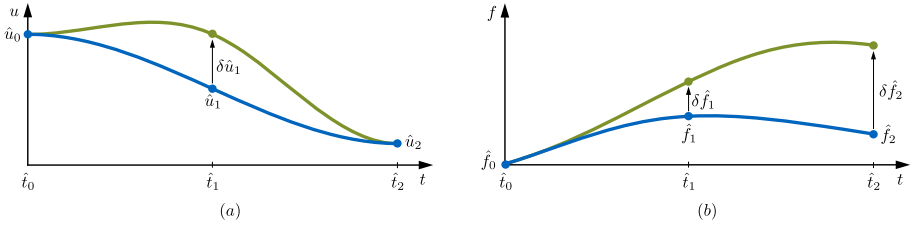


Fig. 2 Influence of changing a control grid node on the reaction force: (a) continuous control function with grid nodes, (b) reaction force due to the control in (a)

shown in Fig. (2), where the change of the control grid node $\delta \hat{u}_1$ leads to a change of $\delta \hat{f}_1$ and $\delta \hat{f}_2$.

The sensitivity analysis uses the following set of parameters: the mass $m = 1$ kg, the damping coefficient $d = 0.5$ Ns/m, the stiffness $c = 1$ N/m, the constant time-integration step size $\Delta t = 0.001$ s, and the final time $t_f = 2$ s. In addition, the control u is discretized by three grid nodes ($M = 2$ uniform distributed intervals in the time interval $[t_0, t_f]$) set to $\hat{\mathbf{u}} = (10, 6, 2)^T$. The initial conditions of the state variables are defined by $\mathbf{x}_0 = \mathbf{0}$, i.e., the initial reaction force is zero.

5.1.1 Discrete adjoint gradient computation

The sensitivities of $d\hat{\mathbf{h}}/dz$ are obtained by using the proposed discrete adjoint gradient in (21), which includes the discrete adjoint variables \mathbf{R} defined in (17)–(19). Referring to the procedure in Sect. 3.4, the first step of the adjoint gradient computation is the symbolic differentiation of constraints. In this example, the reaction force is interpreted as an inequality constraint, and the derivative with respect to the state variables is given by $\frac{\partial \mathbf{h}}{\partial \mathbf{x}} = (c, d)$. Derivatives with respect to ξ are not defined since the state equations are not parameterized. The second step of the adjoint gradient computation is the symbolic differentiation of the state equations. The derivatives with respect to the state variables and the control are represented by

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{pmatrix} 0 & 1 \\ -\frac{c}{m} & -\frac{d}{m} \end{pmatrix} := \mathbf{A} \quad \text{and} \quad \frac{\partial \mathbf{f}}{\partial u} = \begin{pmatrix} 0 \\ \frac{1}{m} \end{pmatrix}, \tag{46}$$

respectively. Note that the simple structure of the linear state equations leads to constant system derivatives and, therefore, no forward integration is required to evaluate the derivatives.

The computation of the discrete adjoint variables is performed in a backward manner in which the final value defined in (17) is given in this example by

$$\mathbf{R}_N = - \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ c & d \end{pmatrix}, \tag{47}$$

where the derivative of the reaction force with respect to state variables is recognized. The explicit Euler method is used to solve the forward dynamics, and, therefore, the specific sensitivity of the right-hand side \mathbf{f} is required, as shown in Sect. 3.2. Substituting (46) and (23) into (18) leads to the algebraic equations of the discrete adjoint variables. Starting from \mathbf{R}_N and proceeding with

$$\mathbf{R}_i = \mathbf{R}_{i+1} (\mathbf{I} + \Delta t \mathbf{A}), \tag{48}$$

the discrete adjoint variables are computed in a backward manner. Note that (48) is defined within the macro time mesh $]\hat{t}_j, \hat{t}_{j+1}[$, $j \in \{0, \dots, M - 1\}$. The discrete adjoint variables at the macro time mesh \hat{t}_j , $j \in \{1, \dots, M\}$ are determined by the intermediate condition in (19) given by

$$\mathbf{R}_{ej} = \mathbf{R}_{ej+1} (\mathbf{I} + \Delta t \mathbf{A}) - \mathbf{B}_{2,j} (c, d). \tag{49}$$

After computing the adjoint variables backward in time, the sensitivity of the inequality constraints $\hat{\mathbf{h}}$ with respect to the grid nodes $\hat{\mathbf{u}}$ can be evaluated by the proposed formulation in (21).

5.1.2 Interpretation

Figure 3 represents the evolution of each component of the adjoint variables \mathbf{R} . The adjoint variables $\mathbf{R} \in \mathbb{R}^{(p+q) \times n}$ consists of six components with $p = 0$, $q = r(M + 1) = 3$, and $n = 2$. It can be observed that the adjoint variables are not necessarily smooth functions due to the intermediate condition in (49). The intermediate condition computes the adjoint variables in the same manner as in (48) and initializes the adjoint variables for the time interval $]\hat{t}_0, \hat{t}_j]$ in addition. The gradient computation of $d\hat{h}_j/d\hat{\mathbf{u}}$ evaluated at the time \hat{t}_j is not influenced by the system dynamics in the time interval $]\hat{t}_j, \hat{t}_M]$ and, therefore, $\mathbf{R}_{j,\bullet} = \mathbf{0}$ in this interval for all \bullet columns of the adjoint variables. To be more precise, the system dynamics of the gradient computation for \hat{h}_j is only taken into account for the time interval $]\hat{t}_0, \hat{t}_j]$, i.e., the

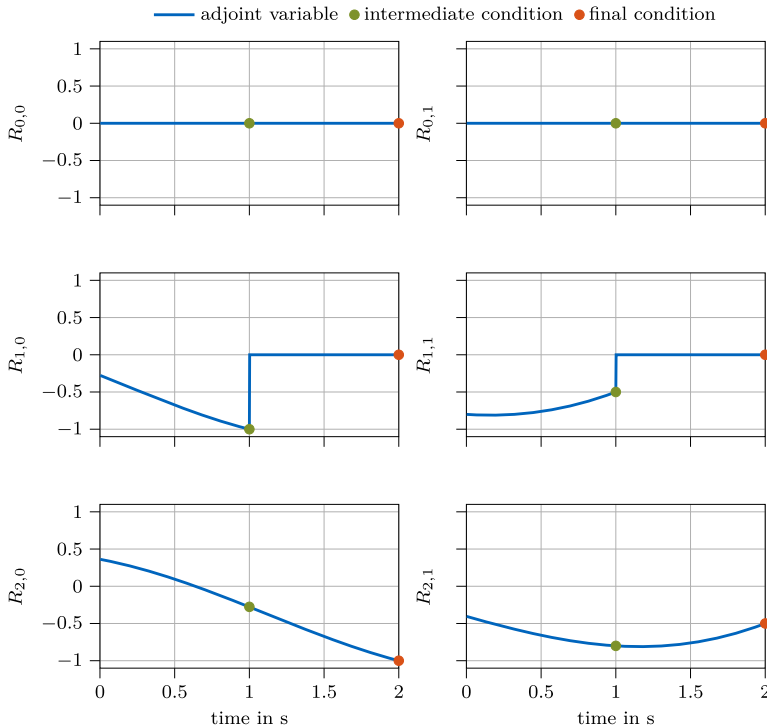


Fig. 3 Time evolution of the discrete adjoint variables

adjoint variables for the time interval $[\hat{t}_j, \hat{t}_M]$ are zero and do not influence the gradient. This effect is seen in the second row of the adjoint variables visualized in Fig. 3. Here, the gradient computation of $d\hat{h}_1/d\hat{\mathbf{u}}$ evaluated at the time $\hat{t}_1 = 1$ s is not influenced by the system dynamics in the time interval $[\hat{t}_1, \hat{t}_2]$ and, therefore, the adjoint variables $R_{1,0}$ and $R_{1,1}$ are zero in this time interval. However, the discrete adjoint sensitivities are compared with the sensitivities computed by the finite-difference method to verify the proposed approach and its implementation. The sensitivity analysis results obtained by both approaches are in good agreement.

5.2 Energy optimal control of a nonlinear spring pendulum

The second example is focused on the energy optimal control of a nonlinear spring pendulum, as depicted in Fig. 4, inspired by the example studied in [1]. The aim is to compute a control to manipulate the mechanical system from an initial to a final state. The example studied in [1] is adapted with an additional inequality constraint during the maneuver to test the proposed discrete adjoint gradient approach. The mechanical system consists of three degrees of freedom $\mathbf{q} = (r_x, r_y, r_z)^T$ describing the position vector of a mass m formulated in the inertial Cartesian coordinate system. The mass is connected to the ground by a spring c , and the strain is formulated by using a Green–Lagrangian-type strain measure

$$\varepsilon = \frac{1}{2l_0^2} (\mathbf{q}^T \mathbf{q} - l_0^2), \tag{50}$$

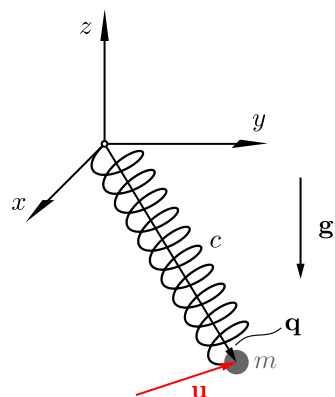
wherein l_0 denotes the strain-free spring length. The mass matrix \mathbf{M} of the mechanical system is defined by using the kinetic energy

$$T = \frac{1}{2} \dot{\mathbf{q}}^T \underbrace{m\mathbf{I}}_{\mathbf{M}} \dot{\mathbf{q}}, \tag{51}$$

where \mathbf{I} is the identity matrix. Forces introduced due to the gravitational acceleration g and the deformation of the spring are defined by

$$\mathbf{Q}_{nl} = \left(\frac{\partial V}{\partial \mathbf{q}} \right)^T. \tag{52}$$

Fig. 4 Nonlinear spring pendulum in a general configuration



The potential energy of the mechanical system reads

$$V = mgr_z + \frac{1}{2}cl_0^2\varepsilon^2, \tag{53}$$

where the first term corresponds to gravity and the second to the deformation of the spring. Note that the force term \mathbf{Q}_{nl} is nonlinear due to the nonlinear Green–Lagrange strain measure in (50). In addition, the mass is driven by a time-dependent control $\mathbf{u} = (u_x, u_y, u_z)^T$ and the corresponding force in the inertial Cartesian coordinate system is defined by the principle of virtual work as $\mathbf{Q}_u = \mathbf{u}$. The state equations are given by a nonlinear first-order differential system

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) = \left(\begin{array}{c} \mathbf{v} \\ \mathbf{M}^{-1}(\mathbf{Q}_u - \mathbf{Q}_{nl}) \end{array} \right), \tag{54}$$

wherein the state variables are expressed by introducing the velocity $\dot{\mathbf{q}} = \mathbf{v}$ of the mass by $\mathbf{x} = (\mathbf{q}, \mathbf{v})^T$. In this example, the control is formulated as proposed by Lichtenecker et al. [22] with $\mathbf{u} = \mathbf{C}\bar{\mathbf{u}}$, where $\mathbf{C} \in \mathbb{R}^{3 \times 3k}$ is a time-dependent cubic spline interpolation matrix and $\bar{\mathbf{u}}^T = (\hat{\mathbf{u}}_x^T, \hat{\mathbf{u}}_y^T, \hat{\mathbf{u}}_z^T) \in \mathbb{R}^{3k}$ is a set of concatenated control grid nodes regarding the control functions. Each control is discretized with k grid nodes defined at a uniformly distributed time mesh within the time interval $[t_0, t_f]$.

According to the example studied in [1], the following set of parameters is used: the mass $m = 1$ kg, the stiffness $c = 0.6$ N/m, the strain-free spring length $l_0 = 5$ m, and the gravitational acceleration $g = 9.81$ m/s². The state variables \mathbf{x} are time-integrated using the explicit Euler scheme on the micro time mesh in the interval $[t_0, t_f]$, where the final time is $t_f = 5$ s with a constant time-integration step size $\Delta t = 0.001$ s. The initial state variables are defined by $\mathbf{x}_0 = (-2, -5, -5, -3, 0, 0)^T$.

5.2.1 Optimization problem

The energy optimal control problem of mechanical systems has been studied by various authors with different formulations of the cost function. In this example, the energy optimal control problem is formulated by minimizing the signal energy required to manipulate the mechanical system. The signal energy is defined as the integrated quadratic control in the time interval $[t_0, t_f]$. To address the energy optimal control problem, the concatenated set of grid nodes is used to define optimization variables, i.e., $\mathbf{z} = \bar{\mathbf{u}} \in \mathbb{R}^{3k}$.

The energy optimal control problem yields the NLP problem formulated as a direct single shooting by

$$\min_{\mathbf{z}} J = \frac{1}{2} \int_{t_0}^{t_f} \mathbf{u}^T \mathbf{u} dt = \frac{1}{2} \mathbf{z}^T \underbrace{\int_{t_0}^{t_f} \mathbf{C}^T \mathbf{C} dt}_{\mathcal{A}} \mathbf{z} \tag{55}$$

s.t.

$$\mathbf{z}_{\min} \leq \mathbf{z} \leq \mathbf{z}_{\max} \tag{56}$$

$$\mathbf{q}_N = \mathbf{q}_f \tag{57}$$

$$\dot{\mathbf{q}}_N = \mathbf{0} \tag{58}$$

$$\hat{\mathbf{l}}/l_{\max} \leq 1 \tag{59}$$

$$\mathbf{x}_{i+1} = \tilde{\mathbf{f}}(\mathbf{x}_i, \mathbf{u}_i), \tag{60}$$

where the control in (55) is formulated with $\mathbf{u} = \mathbf{Cz}$. The position \mathbf{q}_N and velocity $\dot{\mathbf{q}}_N$ are prescribed at the fixed final time t_f . The final position of the mass is defined according to [1] by $\mathbf{q}_f = (2, -10, -4)^T$. In addition, the optimization problem concerns normalized inequality constraints regarding the spring length $l = \sqrt{\mathbf{q}^T \mathbf{q}}$. The inequality constraints are considered at the uniformly distributed macro time mesh $\hat{t}_j, j \in \{0, \dots, M\}$ within the time interval $[t_0, t_f]$. Evaluating the spring length $(M + 1)$ times leads to the concatenated vector $\hat{\mathbf{l}} \in \mathbb{R}^{M+1}$ with $M = 500$. The defined inequality constraints forces the mass to a position within a sphere with radius $r = l_{\max}$ centered on the inertial Cartesian coordinate system with $l_{\max} = 12$ m. Lower and upper bounds of the optimization variables are taken into account with $-5.5 \text{ N} \leq \hat{\mathbf{u}} \leq 5.5 \text{ N}$. As an initial guess, the optimization variables are set to $\mathbf{z} = \mathbf{0}$.

For efficient numerical computation, the NLP is solved with IPOPT 3.14.12 [45] (HSL MA97 to solve linear subproblems). The cost function, constraints, and the respective first-order gradients are provided to IPOPT via an interface. The first-order gradients of the cost function (55) can be easily computed by symbolic differentiation

$$\frac{dJ}{dz} = \mathbf{z}^T \mathcal{A}, \tag{61}$$

where the symmetric property of \mathcal{A} due to the block diagonal matrix \mathbf{C} is utilized. The first-order gradients of the constraints (57)–(59) are computed following the proposed procedure in Sect. 3.4, implemented in Python.

5.2.2 Optimization results

Each control is discretized with $k = 10$ grid nodes leading to $z = 30$ optimization variables. Figure 5 shows the time evolution of the state variables obtained for the energy optimal control. The results are in accordance with the defined equality constraints in (57) and (58) at the final time. Figure 6 visualizes the obtained energy optimal control history and the inequality constraint regarding the spring length. It can be observed that the control variables are within the lower and upper bounds, while the inequality constraint in (59) is active. Similar to the previous example, the discrete adjoint gradients of the constraints are compared with the gradients computed by the finite-difference method to verify the proposed approach and its implementation. The sensitivity analysis results obtained by both approaches are in good agreement.

To demonstrate the efficiency of the proposed discrete adjoint gradient approach, the NLP problem is solved by providing gradients of the constraints to IPOPT once using the

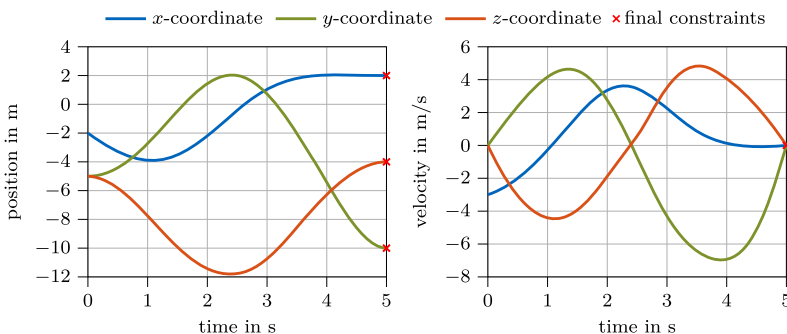


Fig. 5 Time evolution of the state variables obtained for the energy optimal control problem

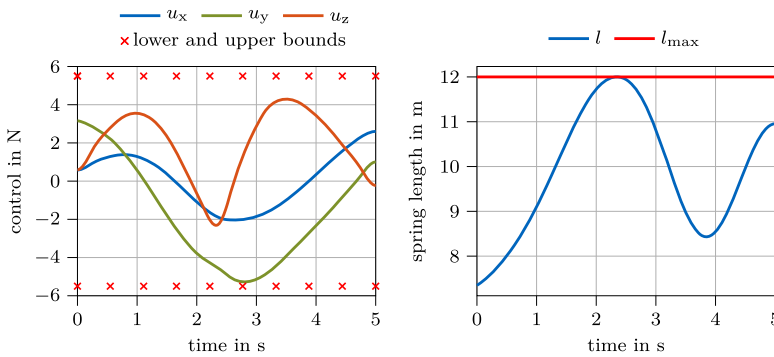


Fig. 6 Optimal control history for the energy optimal control problem and the resulting spring length

Table 1 Runtime comparison of two different approaches to provide first-order gradients of constraints to IPOPT for converged solutions

Grid nodes per control	Type of gradient computation	Runtime
$k = 10$	forward finite-difference	22.3 s
	discrete adjoint method	13.2 s
$k = 20$	forward finite-difference	37.8 s
	discrete adjoint method	14.7 s

finite-difference method and once using the discrete adjoint gradient approach. The runtimes required for converged solutions are given in Table 1. The comparison is performed with $k = 10$ and $k = 20$ grid nodes for each control. The proposed discrete adjoint approach outperforms the finite-difference method in terms of runtimes. Employing the discrete adjoint gradient approach, the change in the runtime required for $k = 10$ and $k = 20$ is small. In contrast, the runtime required for the finite-difference method depends strongly on the number of grid nodes. Note that the number of grid nodes k influences the number of optimization variables $z = 3k$ and, therefore, the runtime for the finite-difference method. The number of iterations to fulfill the KKT conditions is equal for both approaches, which demonstrates the correct implementation of the discrete adjoint gradient approach.

5.3 Optimal control and design of a flexible SCARA

The third example focuses on the combined optimal control and structural optimization problem for flexible multibody systems. The idea of coupling both optimization tasks is promising to obtain the best possible mechanical structure concerning an optimal control problem. Engineers usually do not address the combined structural optimization and optimal control problem; the two challenges are typically considered independently. In addition, multibody systems with flexible components are usually underactuated, and the optimal control problem becomes more complicated than fully actuated systems [39]. Lichtenecker et al. studied in [22] the time-optimal control problem of a SCARA with flexible components. A similar system configuration is used in this example to employ the proposed discrete adjoint approach for the sensitivity analysis of a combined optimal control and structural optimization problem. Thus, a combined set of discrete adjoint gradients is used to efficiently and accurately compute first-order gradients to speed up the runtime in a direct optimization method.

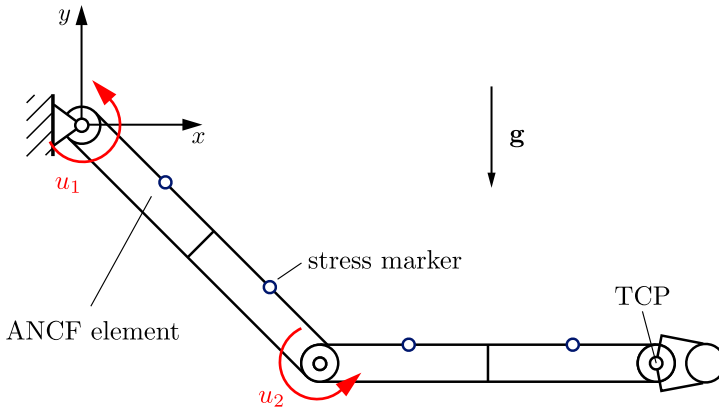


Fig. 7 SCARA in a general undeformed configuration

Inspired by the example studied in [35], the two-arm robot depicted in Fig. 7 is analyzed in the combined optimization problem. The SCARA is driven by two controls, u_1 and u_2 , in which the control variables are formulated by cubic splines similar to the previous example. In application to industrial robots, smooth trajectory planning is essential and has been presented using cubic splines, e.g., in [8, 37].

Each arm is divided into two ANCF elements, and an additional mass is attached to the tool-center point (TCP). Moreover, a stress marker is attached to each structural element to determine the equivalent stress σ_v while the robot performs a task. The material properties of the structural elements are set to $E = 3e9 \text{ N/m}^2$ and $\rho = 1300 \text{ kg/m}^3$ for the Young's modulus and the density, respectively. The length of both arms is $l_1 = l_2 = 1 \text{ m}$ and the viscous damping coefficient in the revolute joints is set to $d_1 = d_2 = 0.2 \text{ Nm s/rad}$. The width of the structural elements is set to $w^{(e)} = 0.002 \text{ m}$. In addition, the system is affected by the gravity field, and an additional mass $m_E = 1 \text{ kg}$ is attached to the TCP. The state variables \mathbf{x} are integrated using the implicit Euler scheme on the micro time mesh in the interval $[t_0, t_f]$, where the final time is $t_f = 2 \text{ s}$ with a constant time-integration step size $\Delta t = 0.001 \text{ s}$.

5.3.1 Optimization problem

Structural-optimization problems can be treated with so-called weakly and fully coupled methods [42]. Weakly coupled methods are based on equivalent static loads, while fully coupled methods incorporate the system dynamics into the optimization process. This example focuses on extending fully coupled methods to embed the optimal control of flexible multibody systems. Considering an optimal control problem and a structural optimization problem leads to a combined set of optimization variables, including the control parameterization and design parameter of the multibody system. In this example, the structural elements are parameterized by the height $h^{(e)}$, while the width $w^{(e)}$ and the length $l^{(e)}$ are set to constant values. The set of design parameters regarding the multibody system depicted in Fig. 7 is defined by $\xi^T = (h^{(1)}, h^{(2)}, h^{(3)}, h^{(4)})$. The optimal control problem of flexible multibody systems requires smooth control functions to reduce vibrations. In this example, a continuity requirement up to C^2 of the control is enforced, similar to the previous example in Sect. 5.2, by a cubic spline interpolation. Both continuous control functions u_1 and u_2 are discretized with $k = 21$ grid nodes at a uniformly distributed time mesh within the time

interval $[t_0, t_f]$. The discretization of the control leads to the set of grid nodes $\bar{\mathbf{u}}^\top = (\hat{\mathbf{u}}_1^\top, \hat{\mathbf{u}}_2^\top)$. Concatenating the parameterization of the multibody system and the control leads to the combined set of optimization variables $\mathbf{z}^\top = (\boldsymbol{\xi}^\top, \bar{\mathbf{u}}^\top)$.

Minimizing the mass of mechanical systems is a common approach in structural optimization to enable innovative lightweight designs. In this paper, the mass minimization yields the NLP problem formulated as direct single shooting by

$$\min_{\mathbf{z}} m = \rho \sum_{(e)} w^{(e)} h^{(e)} l^{(e)} \tag{62}$$

s.t.

$$\mathbf{z}_{\min} \leq \mathbf{z} \leq \mathbf{z}_{\max} \tag{63}$$

$$\mathbf{r}_{\text{TCP},N} = \mathbf{r}_f \tag{64}$$

$$\dot{\mathbf{r}}_{\text{TCP},N} = \mathbf{0} \tag{65}$$

$$\hat{\sigma}_V / \sigma_{V\max} \leq 1 \tag{66}$$

$$\mathbf{x}_{i+1} = \tilde{\mathbf{f}}(\mathbf{x}_i, \mathbf{x}_{i+1}, \mathbf{u}_i, \mathbf{u}_{i+1}, \boldsymbol{\xi}), \tag{67}$$

with special attention to position and velocity constraints of the TCP, $\mathbf{r}_{\text{TCP},N}$, and $\dot{\mathbf{r}}_{\text{TCP},N}$, respectively, at the fixed final time t_f . The final position of the TCP is defined by $\mathbf{r}_f = (1, 1)^\top$. In addition, the optimization problem concerns normalized inequality constraints regarding the equivalent stress σ_V of the four markers attached to the structure. All four markers are evaluated at the macro time mesh $\hat{t}_j, j \in \{0, \dots, M\}$ with $M = 100$ leading to the concatenated equivalent stress vector $\hat{\sigma}_V$. The upper limit of the equivalent stress is $\sigma_{V\max} = 1.1e7 \text{ N/m}^2$. Lower and upper bounds of the optimization variables are taken into account with $0.002 \text{ m} \leq h^{(e)} \leq 0.02 \text{ m}$ and $-5 \text{ Nm} \leq \hat{u} \leq 5 \text{ Nm}$, respectively. Regarding the initial conditions of the state variables, the robot is defined in the undeformed configuration, where both arms are hanging vertically downward with generalized velocities equal to zero.

In terms of initializing the NLP problem, a two-stage procedure is utilized with decoupling of the optimization variables. The first stage solves an optimal control problem with a constant height of the structural elements, i.e., the optimization variables $\mathbf{z} = \bar{\mathbf{u}}$ consists of the grid nodes. The aim is to identify a control with a defined set of parameters $\boldsymbol{\xi}$ that manipulates the robot from the initial configuration so that the TCP satisfies the constraints (64) and (65) at the final time. As an initial guess for this first-stage optimization, the control grid nodes are set to $\hat{u} = 0 \text{ Nm}$ and the constant height of the elements is defined by $h^{(e)} = 0.02 \text{ m}$. The preoptimized set of control grid nodes $\bar{\mathbf{u}}^*$ and heights of the elements $h^{(e)}$ are employed as an initial guess to the combined optimization problem.

For efficient numerical computation, the NLP is solved with IPOPT 3.14.12 [45] (HSL MA97 to solve linear subproblems). The cost function, constraints, and the respective first-order gradients are provided to IPOPT via an interface. The first-order gradients of the cost function (62) can be easily computed by symbolic differentiation, while the first-order gradients of the constraints (64)–(66) are computed following the proposed procedure in Sect. 3.4, implemented in Python.

5.3.2 Optimization results

Figure 8 shows the obtained control history for both controls. It can be observed that the control is within the lower and upper bounds, while the stress constraint in (66) is active.

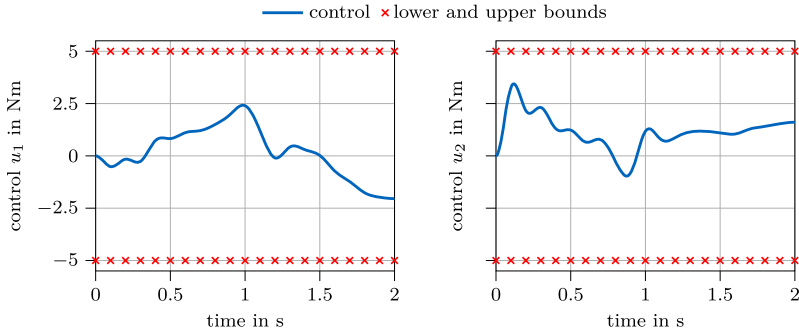


Fig. 8 Optimal control history of a flexible two-arm robot for a rest-to-rest maneuver

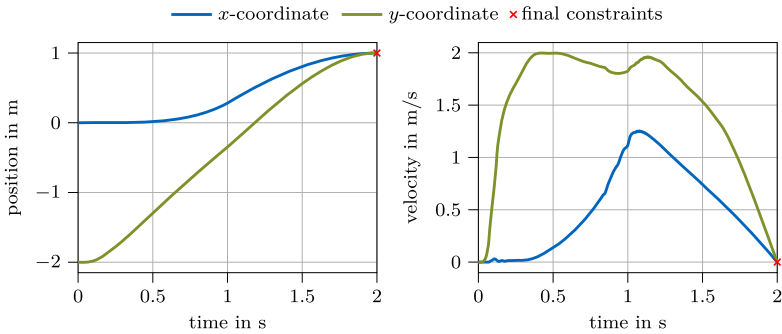


Fig. 9 Time evolution of the position and velocity of the TCP obtained for the optimal control and design problem

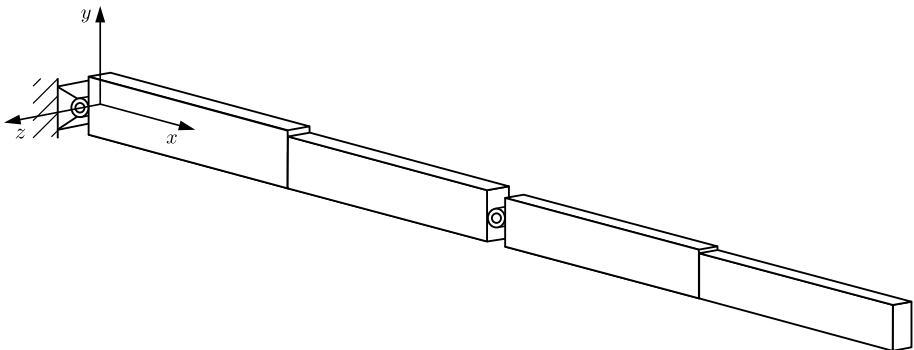
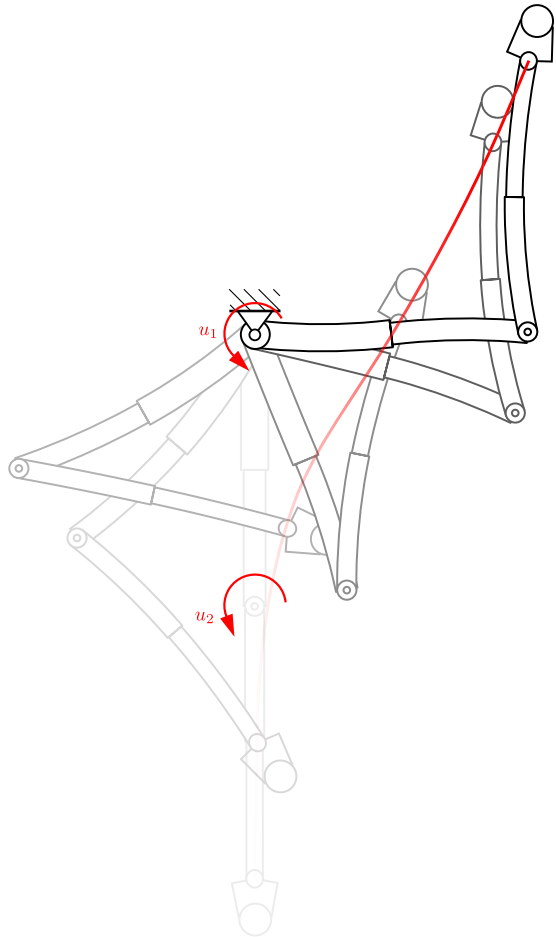


Fig. 10 Final sizing of the flexible two-arm robot

Applying the optimal control to the SCARA leads to the position and velocity of the TCP as shown in Fig. 9. Note there are no acceleration constraints defined at the final time in this example. It can be observed that the equality constraints in (64) and (65) are fulfilled at the final time. Figure 10 visualizes the final sizing of the structural elements. The final sizing provides a structure in which the height of the structural elements becomes smaller towards

Fig. 11 Snapshots of the robot's motion history



the TCP; thus, the bending stiffness also becomes smaller towards the TCP. The initial mass of the robot $m_{\text{init}} = 0.1040$ kg is reduced by the final design to $m^* = 0.0450$ kg, which corresponds to a reduction of 56.7% regarding the initial mass. With the final design and the corresponding control, the SCARA undergoes a large deformation during manipulation. However, the results are in accordance with the constraints posed in (63)–(67) and provide a local minimum of the robot's mass. Snapshots of the robot's motion are illustrated in Fig. 11.

It has to be mentioned that normalizing inequality constraints is essential since the numerical value of the equality constraints in (64) and (65) are relatively small compared to those of the inequality constraint in (66). Without normalizing, the optimization would not converge to a local minimum. Similar to the previous examples, the discrete adjoint gradients are verified by applying the finite-difference method. The sensitivity analysis results obtained by both approaches are in good agreement. The proposed discrete adjoint gradient approach is an efficient technique to incorporate a large number of optimization variables, and the computational effort is less than using the finite-difference method. A comparison of the runtimes required to compute first-order gradients of the constraints is performed to demonstrate the efficiency of the proposed discrete adjoint gradient approach. The gradients

are computed with the optimal set of optimization variables \mathbf{z}^* ten times in a row. The average runtime required for the finite-difference method is $t_{\text{FDM}} = 592.4$ s, while the average runtime required for the discrete adjoint method is $t_{\text{DAM}} = 52.2$ s. The proposed approach reduces the computational time by 91.2% regarding the finite-difference method. The high computational time in the finite-difference method is because the number of state equations to be solved depends on the number of optimization variables. It has to be emphasized that the computation of gradients is required at each iteration of the NLP solver package, which encourages the use of the proposed discrete adjoint gradient method to improve computational efficiency.

6 Conclusion

This paper discusses adjoint-based sensitivity analysis for dynamic systems in gradient-based optimization problems. Deriving adjoint gradients is mathematically more laborious than simply computing finite differences for numerical gradients. However, the significant time advantage when using adjoint gradients for the sensitivity analysis justifies the considerable preprocessing effort. This paper presents a novel discrete adjoint gradient approach to incorporate (in)equality constraints. Moreover, the paper shows the application of different time-integration schemes, highlighting their efficiency and applicability to large-scale problems. Three numerical examples are investigated to show the application of the proposed discrete adjoint gradient approach. The sensitivity analysis of an academic example discusses the role of the discrete adjoint variables. The energy optimal control problem of a nonlinear spring pendulum studies the efficiency of the proposed approach. In addition, the proposed discrete adjoint gradients are utilized in a coupled optimal control and optimal design problem in flexible multibody dynamics.

Acknowledgements Daniel Lichtenecker and Karin Nachbagauer acknowledge support from the Technical University of Munich – Institute for Advanced Study.

Author contributions D. Lichtenecker performed simulation and analysis, and wrote the main manuscript text. K. Nachbagauer supervised the project. Both authors jointly conceived the presented ideas. Both authors reviewed the manuscript.

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

Competing interests The authors declare no competing interests.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Betsch, P., Becker, C.: Conservation of generalized momentum maps in mechanical optimal control problems with symmetry. *Int. J. Numer. Methods Eng.* **111**(2), 144–175 (2017). <https://doi.org/10.1002/nme.5459>

2. Betts, J.T.: Practical Methods for Optimal Control and Estimation Using Nonlinear Programming, 2nd edn. SIAM, Philadelphia (2010). <https://doi.org/10.1137/1.9780898718577>
3. Boopathy, K., Kennedy, G.: Adjoint-based derivative evaluation methods for flexible multibody systems with rotorcraft applications. In: 55th AIAA Aerospace Sciences Meeting (2017). <https://doi.org/10.2514/6.2017-1671>
4. Bryson, A.E., Ho, Y.C.: Applied Optimal Control: Optimization, Estimation, and Control. Taylor & Francis, New York (1975). <https://doi.org/10.1201/9781315137667>
5. Butcher, J.C.: Numerical Methods for Ordinary Differential Equations. Wiley, New York (2016). <https://doi.org/10.1002/9781119121534>
6. Callejo, A., Sonnevill, V., Bauchau, O.A.: Discrete adjoint method for the sensitivity analysis of flexible multibody systems. *J. Comput. Nonlinear Dyn.* **14**(2), 021001 (2019). <https://doi.org/10.1115/1.4041237>
7. Cao, Y., Li, S., Petzold, L., Serban, R.: Adjoint sensitivity analysis for differential-algebraic equations: the adjoint DAE system and its numerical solution. *SIAM J. Sci. Comput.* **24**(3), 1076–1089 (2003). <https://doi.org/10.1137/S1064827501380630>
8. Constantinescu, D., Croft, E.A.: Smooth and time-optimal trajectory planning for industrial manipulators along specified paths. *J. Robot. Syst.* **17**(5), 233–249 (2000). [https://doi.org/10.1002/\(SICI\)1097-4563\(200005\)17:5<233::AID-ROB1>3.0.CO;2-Y](https://doi.org/10.1002/(SICI)1097-4563(200005)17:5<233::AID-ROB1>3.0.CO;2-Y)
9. Della Santina, C., Duriez, C., Rus, D.: Model-based control of soft robots: a survey of the state of the art and open challenges. *IEEE Control Syst. Mag.* **43**(3), 30–65 (2023). <https://doi.org/10.1109/MCS.2023.3253419>
10. Eberhard, P.: Adjoint variable method for sensitivity analysis of multibody systems interpreted as a continuous, hybrid form of automatic differentiation. In: Proceedings of the Second International Workshop of Computational Differentiation, pp. 12–14 (1996)
11. 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). <https://doi.org/10.1016/j.compstruc.2018.12.007>
12. Gufler, V., Wehrle, E., Zwölfer, A.: A review of flexible multibody dynamics for gradient-based design optimization. *Multibody Syst. Dyn.* **53**(4), 379–409 (2021). <https://doi.org/10.1007/s11044-021-09802-z>
13. Haug, E.J., Wehage, R.A., Mani, N.K.: Design sensitivity analysis of large-scale constrained dynamic mechanical systems. *J. Mech. Transm. Autom. Des.* **106**(2), 156–162 (1984). <https://doi.org/10.1115/1.3258573>
14. Hawkes, E.W., Majidi, C., Tolley, M.T.: Hard questions for soft robotics. *Sci. Robot.* **6**(53), eabg6049 (2021). <https://doi.org/10.1126/scirobotics.abg6049>
15. Held, A., Seifried, R.: Gradient-based optimization of flexible multibody systems using the absolute nodal coordinate formulation. In: Proceedings of the 6th ECCOMAS Thematic Conference on Multibody Dynamics (2013)
16. Held, A., Seifried, R.: Adjoint sensitivity analysis of multibody system equations in state-space representation obtained by QR decomposition. In: Proceedings of the 11th ECCOMAS Thematic Conference on Multibody Dynamics (2023)
17. Johnston, L., Patel, V.: Second-order sensitivity methods for robustly training recurrent neural network models. *IEEE Control Syst. Lett.* **5**(2), 529–534 (2021). <https://doi.org/10.1109/LCSYS.2020.3001498>
18. Karush, W.: Minima of functions of several variables with inequalities as side constraints. Master's thesis, Department of Mathematics, University of Chicago (1939)
19. Kuhn, H.W., Tucker, A.W.: Nonlinear programming. In: Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, pp. 481–492 (1951)
20. Lam, S.K., Pitrou, A., Seibert, S.: Numba: a LLVM-based python JIT compiler. In: Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC. Assoc. Comput. Mach., New York (2015). <https://doi.org/10.1145/2833157.2833162>
21. Lauß, T., Oberpeilsteiner, S., Steiner, W., Nachbagauer, K.: The discrete adjoint method for parameter identification in multibody system dynamics. *Multibody Syst. Dyn.* **42**(4), 397–410 (2018). <https://doi.org/10.1007/s11044-017-9600-9>
22. Lichtenecker, D., Rixen, D., Eichmeir, P., Nachbagauer, K.: On the use of adjoint gradients for time-optimal control problems regarding a discrete control parameterization. *Multibody Syst. Dyn.* **59**(3), 313–334 (2023). <https://doi.org/10.1007/s11044-023-09898-5>
23. Lichtenecker, D., Eichmeir, P., Nachbagauer, K.: On the usage of analytically computed adjoint gradients in a direct optimization for time-optimal control problems. In: Optimal Design and Control of Multibody Systems. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-50000-8_14
24. Lions, J.L.: Optimal Control of Systems Governed by Partial Differential Equations. Springer, New York (1971)

25. López Varela, Á., Sandu, C., Sandu, A., Dopico Dopico, D.: Discrete adjoint variable method for the sensitivity analysis of ALI3-P formulations. *Multibody Syst. Dyn.* (2023). <https://doi.org/10.1007/s11044-023-09911-x>
26. Maciąg, P., Malczyk, P., Frączek, J.: Hamiltonian direct differentiation and adjoint approaches for multi-body system sensitivity analysis. *Int. J. Numer. Methods Eng.* **121**(22), 5082–5100 (2020). <https://doi.org/10.1002/nme.6512>
27. Maciąg, P., Malczyk, P., Frączek, J.: Adjoint-based feedforward control of two-degree-of-freedom planar robot. In: *Proceedings of the 11th ECCOMAS Thematic Conference on Multibody Dynamics* (2023)
28. Marler, R.T., Arora, J.S.: Survey of multi-objective optimization methods for engineering. *Struct. Multi-discip. Optim.* **26**(6), 369–395 (2004). <https://doi.org/10.1007/s00158-003-0368-6>
29. Martins, J., Hwang, J.T.: Review and unification of methods for computing derivatives of multidisciplinary computational models. *AIAA J.* **51**(11), 2582–2599 (2013). <https://doi.org/10.2514/1.J052184>
30. Meurer, A., Smith, C.P., Paprocki, M., Čertík, O., Kirpichev, S.B., Rocklin, M., Kumar, A., Ivanov, S., Moore, J.K., Singh, S., Rathnayake, T., Vig, S., Granger, B.E., Muller, R.P., Bonazzi, F., Gupta, H., Vats, S., Johansson, F., Pedregosa, F., Curry, M.J., Terrel, A.R., Roučka, Š., Saboo, A., Fernando, I., Kulal, S., Cimrman, R., Scopatz, A.: SymPy: symbolic computing in Python. *PeerJ Comput. Sci.* **3**, e103 (2017). <https://doi.org/10.7717/peerj-cs.103>
31. Nachbagauer, K., Oberpeilsteiner, S., Sherif, K., Steiner, W.: The use of the adjoint method for solving typical optimization problems in multibody dynamics. *J. Comput. Nonlinear Dyn.* **10**(6), 061011 (2015). <https://doi.org/10.1115/1.4028417>
32. Nadarajah, S., Jameson, A.: A comparison of the continuous and discrete adjoint approach to automatic aerodynamic optimization. In: *38th Aerospace Sciences Meeting and Exhibit* (2000). <https://doi.org/10.2514/6.2000-667>
33. Nocedal, J., Wright, S.J.: *Numerical Optimization*, 2nd edn. Springer, New York (2006). <https://doi.org/10.1007/978-0-387-40065-5>
34. Omar, M.A., Shabana, A.A.: A two-dimensional shear deformable beam for large rotation and deformation problems. *J. Sound Vib.* **243**(3), 565–576 (2001). <https://doi.org/10.1006/jsvi.2000.3416>
35. Oral, S., Kemal Ider, S.: Optimum design of high-speed flexible robotic arms with dynamic behavior constraints. *Comput. Struct.* **65**(2), 255–259 (1997). [https://doi.org/10.1016/S0045-7949\(96\)00269-6](https://doi.org/10.1016/S0045-7949(96)00269-6)
36. Rackauckas, C., Ma, Y., Martensen, J., Warner, C., Zubov, K., Supekar, R., Skinner, D., Ramadhan, A., Edelman, A.: *Universal Differential Equations for Scientific Machine Learning* (2020) <https://doi.org/10.48550/arXiv.2001.04385>. [arXiv:2001.04385](https://arxiv.org/abs/2001.04385). ArXiv preprint
37. Reiter, A., Müller, A., Gattringer, H.: On higher order inverse kinematics methods in time-optimal trajectory planning for kinematically redundant manipulators. *IEEE Trans. Ind. Inform.* **14**(4), 1681–1690 (2018). <https://doi.org/10.1109/TII.2018.2792002>
38. Schneider, S., Betsch, P.: On optimal control problems in redundant coordinates. In: *Proceedings of the 11th ECCOMAS Thematic Conference on Multibody Dynamics* (2023)
39. Seifried, R.: *Dynamics of underactuated multibody systems: modeling, control and optimal design*. In: *Solid Mechanics and Its Applications*. Springer, Berlin (2014). <https://doi.org/10.1007/978-3-319-01228-5>
40. Shabana, A.A.: Definition of the slopes and the finite element absolute nodal coordinate formulation. *Multibody Syst. Dyn.* **1**(3), 339–348 (1997). <https://doi.org/10.1023/A:1009740800463>
41. Sun, J., Tian, Q., Hu, H.: Structural optimization of flexible components in a flexible multibody system modeled via ANCF. *Mech. Mach. Theory* **104**, 59–80 (2016). <https://doi.org/10.1016/j.mechmachtheory.2016.05.008>
42. Tromme, E., Brüls, O., Duysinx, P.: Weakly and fully coupled methods for structural optimization of flexible mechanisms. *Multibody Syst. Dyn.* **38**(4), 391–417 (2016). <https://doi.org/10.1007/s11044-015-9493-4>
43. Tromme, E., Held, A., Duysinx, P., Brüls, O.: System-based approaches for structural optimization of flexible mechanisms. *Arch. Comput. Methods Eng.* **25**(3), 817–844 (2018). <https://doi.org/10.1007/s11831-017-9215-6>
44. Vanpaemel, S., Vermaut, M., Desmet, W., Naets, F.: Input optimization for flexible multibody systems using the adjoint variable method and the flexible natural coordinates formulation. *Multibody Syst. Dyn.* **57**(3), 259–277 (2023). <https://doi.org/10.1007/s11044-023-09874-z>
45. Wächter, A., Biegler, L.T.: On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math. Program.* **106**(1), 15–57 (2006). <https://doi.org/10.1007/s10107-004-0559-y>