Nonlinear Dyn (2023) 111:11297–11313
https://doi.org/10.1007/s11071-023-08433-7

ORIGINAL PAPER

Recursive inverse dynamics sensitivity analysis of open-tree-type multibody systems

Altay Zhakatayev · Yuriy Rogovchenko · Matthias Pätzold

Received: 14 October 2022 / Accepted: 17 March 2023 / Published online: 13 April 2023

Abstract We present a first-order recursive approach to sensitivity analysis based on the application of the direct differentiation method to the inverse Lagrangian dynamics of rigid multibody systems. Our method is simple and efficient and is characterized by the following features. Firstly, it describes the kinematics of multibody systems using branch connectivity graphs and joint-branch connectivity matrices. For most mechanical systems with an open-tree kinematic structure, this method turns out to be more efficient compared to other kinematic descriptions employing joint or link connectivity graphs. Secondly, a recursive sensitivity analysis is presented for a dynamic system with an open-tree kinematic structure and inverse dynamic equations described in terms of the Lagrangian formalism. Thirdly, known approaches to recursive inverse dynamic and sensitivity analyses are modified to include dynamic systems with external forces and torques acting simultaneously at all joints. Finally, the proposed method for sensitivity analysis is easy to implement and computationally efficient. It can be utilized to evaluate the derivatives of the dynamic equations of multibody systems in gradient-based optimization algorithms. It also allows less experienced users to perform sensitivity analyses using the power of high-level programming languages such as MATLAB. To illustrate the method, simulation results for a human body model are discussed. The shortcomings of the method and possible directions for future work are outlined.

Keywords Sensitivity analysis · Multibody systems · Recursive inverse dynamics · Branch connectivity graph

Mathematics Subject Classification 70E55 · 37N05 · 70E60

A. Zhakatayev (✉)
Department of ICT, University of Agder, 4604 Kristiansand, Norway
e-mail: altay.zhakatayev@uia.no

Y. Rogovchenko
Dept. of Mathematics, University of Agder, 4604 Kristiansand, Norway
e-mail: yuriy.rogovchenko@uia.no

M. Pätzold
Department of ICT, University of Agder, 4879 Grimstad, Norway
e-mail: matthias.paetzold@uia.no

1 Introduction

Recent advancements in the analysis of nonlinear dynamics of mechanical systems has been significantly facilitated by the progress in numerical methods, optimal control, nonlinear control, optimal design, system identification, and related fields. Combination of new efficient techniques with a rapidly growing computational power enables researchers to tackle problems deemed unattainable a decade ago. The problem of optimal control of a nonlinear dynamic sys-

tem such as human body is one such example. Our main motivation in this paper is the implementation of the dynamic and sensitivity analysis of the human body.

Equations of motion for multibody systems can be rough-ly divided into two large classes in terms of their mathematical appearance or form: closed-form and recursive equations [24] .[1] In the closed form, also called a *bulk* form, dynamic equations of motion are written as a single vector differential equation containing all generalized coordinates. In the recursive form, all terms in the dynamic equations for generalized coordinates of one joint are presented as functions of generalized coordinates of neighboring joint. The main advantage of the closed form is that is it easier to handle and analyze with analytical tools, primarily for the design of controllers in the state space. However, in this case equations of motion become complicated and highly nonlinear, especially as the degrees of freedom (DOF) of the system increase. The attractive advantages of the recursive formulation are its computational efficiency and the ease of coding. The main disadvantage is the "distortion" of the structure of the dynamic model to the form that excludes the possibility of gaining the insight into the dynamics of the system and exploiting it efficiently [24]. However, some recursive formulations do not have this shortcoming and still allow to gain insight into dynamics [33,48].

Multibody dynamics can be also classified as forward dynamics (kinematic variables are not known) and inverse dynamics (force/torque variables are not known) [12,26,51]. However, in applied problems neither positions and velocities nor input forces and torques are known in advance; these variables should be determined during simulation. The problems where both kinematic and dynamic variables are not known fall into the third category; they are formulated as optimization problems and solved using optimization tools. In various engineering fields this class of problems is called *trajectory optimization*, *motion synthesis*, *motion control,* or *predictive dynamics* [1]. Efficient solution of optimization problems with gradient-based methods depends largely on the gradients of the objective function and constraints. Calculation of gradient matrices and vectors is called sensitivity analysis. In essence, sensitivity analysis is the inverse of the optimization problem. In the optimization problem, the task is to minimize the objective function. This is equivalent to the following problem: given the value for the gradient of the objective function, find the value of the independent variable which satisfies it. In the sensitivity analysis the problem is the opposite: given the value for the independent variable, evaluate the gradient of the objective function. Sensitivity analysis can be performed with respect to states (generalized coordinates, velocities, and accelerations) [1,57]. Sensitivity analysis with respect to physical system parameters (mass, inertia terms, lengths, etc.) or any other design parameters (such as initial conditions) is termed *design sensitivity* [27,53,55]. Sensitivity analysis is important for problems involving design optimization, parameter estimation, and model correlation. It is also crucial for implicit numerical integration of differential-algebraic systems and kinematic workspace analysis of multibody systems [49].

Sensitivity analysis can be performed using analytical, semi-analytical, and numerical methods [10]. Methods for numerical sensitivity analysis can be further classified as finite difference and automatic differentiation methods. Analytical methods include direct differentiation and adjoint variable method [5]. We draw the reader's attention to the fact that the above classification is not standardized and accurate, because there are hybrid methods that combine the aforementioned techniques. For example, there might be direct and/or adjoint variable method combined with numerical and/or automatic differentiation [9,10,37]. The above classification is provided only for information purposes. The finite difference method is commonly employed for evaluating sensitivity matrices. It is simple and easy to implement, but its accuracy depends on the magnitude of the perturbation and is affected by truncation and rounding errors. Therefore, finite difference method may lead to ill-conditioned problems and unreliable results; its computational costs are prohibitively high for large DOF systems [19]. The automatic differentiation method is based on the application of the symbolic chain rule of differentiation; its accuracy and speed are superior to the finite difference method. Open-source software based on automatic differentiation method and written in C++ (DRAKE [52], MBSlib [56], and RobCoGen [45]) can be used

[1] We do not discuss in this paper different formulations of multibody dynamics, such as Newton-Euler, Lagrangian, Hamiltonian, Kane's formulations, Maggi's equations, etc. [38].

to perform sensitivity analysis of dynamic equations with respect to generalized coordinates, velocities, and parameters. The advantages of the automatic differentiation method include possibilities to simulate forward, inverse, and hybrid dynamics, parameter estimations, and trajectory optimization. The main limitation is the need to couple automatic differentiation software with that employed by the user; this might require additional learning. For instance, it is known that the wrong application of the chain rule of differentiation can lead to erroneous results [16]. Furthermore, computation of implicit derivatives often requires costly time integration. The direct differentiation method is based on a straightforward application of differentiation rules in the equations of motion [17,49]. Its advantages include easiness of implementation, accuracy, and higher numerical stability [10]; the method is especially efficient when the number of sensitivity variables is small. Examples of the application of the direct differentiation method for sensitivity analysis of multibody mechanical systems can be found in [6,15,47]. In the adjoint variable method, the objective function is modified with dynamic equations and constraints and additional adjoint conditions that simplify the computation of the Jacobian of the objective function are obtained [28,44]. This method is accurate and numerically efficient, but can become complex and lengthy since it requires backward integration in time [7,46]; it is efficient when the number of objective functions is small. Relevant examples of the application of the adjoint variable method for sensitivity analysis of multibody mechanical systems can be found in [4,7,16,29,44]. One can also find in the literature approaches based on the combination of existing methods. For instance, a semi-analytical sensitivity analysis replaces symbolic derivatives with respect to design variables in analytic expressions by finite differences combining the simplicity of the finite difference method with the accuracy of the adjoint variable and direct differentiation methods in [46].

Most research on the sensitivity analysis uses a closed-form dynamic formulation where all equations are written using a single vector–matrix form [6,7,10,15–18,29,44,46,47,49]. To the best of our knowledge, only a few papers on the recursive sensitivity analysis of mechanical systems with open-tree topology structure are available, although it is often computationally more efficient compared to "bulk" formulations for large multibody systems [5].[2] Closed-form expressions for the sensitivity of multibody system dynamics equations using the spatial notation provided in [25] are abstract and their correct implementation requires substantial user effort. A useful framework for generating motion sequences in musculoskeletal systems was designed by combining the computational tools of MATLAB with the modeling capabilities of OpenSim [39]. However, the sensitivity analysis is not presented there; it was performed numerically using forward finite differences. As expected, the authors noted that the `fmincon` solver was very slow. Sensitivity analysis for simple open-chain multibody structures using inverse recursive Newton-Euler dynamic formulation and spatial notation (as in [21]) was performed in [20]. Using the aggregate force and momentum expressions (changing the order in which terms are computed), linear time analytical first derivatives of the objective function were obtained. However, this method provides true values not for all joint momenta and torques but only for the base joint. We would like to emphasize that the direct differentiation sensitivity analysis using the inverse recursive Lagrangian formulation [1,57], forward recursive Kane's notation [3], and transfer matrix method [54] are valid for simple open-chain structures.

We use Lagrange's formulation of the system dynamics for improving the sensitivity analysis method suggested in [1,57] for open-tree topological structures. Our work complements the recursive sensitivity analysis techniques based on the Kane's method [8,31,43] and on the Newton-Euler method with the spatial notation [33,50]. Contrary to the sensitivity analysis derived in [8,31,43] for forward recursive dynamics, we perform it for inverse recursive dynamics. In comparison with the forward dynamics formulation, the inverse dynamics does not require time integration and increases the efficiency of the sensitivity analysis. The main differences between our proposed method to describe system topology and the method used in [33] are the following. Firstly, we propose a simple method to describe system topology using the branch connectivity graph and the joint-branch connectivity matrix. In contrast, the square adjacency matrix and the block-weighted adjacency matrix are used in [33]. The adjacency matrix carries the same information as

[2] In fact, there are specific "bulk" formulations of multibody system dynamics, which result in very efficient computational codes. For example, using natural Cartesian coordinates [13,36].

the branch connectivity graph and the joint-branch connectivity matrix but requires more space and so is less efficient. The block-weighted adjacency matrix has a larger size than the adjacency matrix. Secondly, in this work, a homogeneous transformation matrix is used rather than the spatial operator algebra notation (6-D vector formulation) as in [33,50]. Our work extends the possibilities of recursive dynamics and sensitivity analysis presented in [1,57] by allowing the inclusion of external forces and torques acting simultaneously at all joints and/or links. This is necessary, for example, for the modeling of two reaction forces acting on both feet during the double support phase of walking. We expect that the proposed sensitivity analysis will reduce the time to evaluate sensitivity information compared to traditional numerical methods. This in turn might reduce the time required by the conventional `fmincon` solver for computations required in optimization problems for nonlinear dynamic systems with many DOF, such as the human body. Since using MATLAB for testing and debugging is easier than, for instance, employing the interior point optimizer IPOPT [39], our contribution should positively impact research on multibody dynamics.

In summary, we present a first-order recursive sensitivity analysis based on the application of the direct differentiation method to the inverse Lagrangian dynamics of rigid multibody systems which is not affected by the perturbations of design variables. Major contributions in this paper are:

- a new method for describing the topology of mechanical systems with an open-tree structure is proposed;

- known recursive dynamic and sensitivity analyses are modified for the use with dynamic systems having an open-tree structure where external forces and torques act simultaneously on all joints;

- the proposed algorithm can be easily implemented in MATLAB thus allowing the use of high-level programming capabilities for the human motion synthesis.

The paper is organized as follows. Section 2 introduces kinematics of the multibody mechanical system. Dynamic equations of motion are defined in Sect. 3, followed by the sensitivity analysis in Sect. 4. Simulation model is presented in Sect. 5 and simulation results are discussed in Sect. 6.

2 Kinematics

2.1 Joint connectivity graph

A mechanical system can be represented as a finite number of connected links and joints. A graph, on the other hand, is a collection of nodes and edges [32]. To correctly represent the kinematics of a mechanical system, the connectivity of the joints should be described. To this end, we utilize the *joint connectivity graph*, an undirected graph similar to the one used in [22,23] where a node represents a link and an arc represents a joint. Since every type of joint (prismatic, revolute, spherical, universal joint, etc.) has to be defined separately, this approach is less efficient in practice. Similarly, each joint type requires its own kinematic description in [33]. Our modification of the connectivity graph employs nodes (vertices) for representing joints, while the arcs (edges) represent connections between neighboring joints. If a mechanical system has an open-tree structure (that is, it does not have closed-loop chains), then its joint connectivity graph is a *joint topological tree*. On a joint connectivity graph, all joints are numbered so that a given joint $i \in \mathbb{N}$ has a lower number than any of its children, and a higher number than any of its parents, see Fig. 1a. In other words, if a number $\lambda_j(i)$ denotes a parent and $\nu_j(i)$ denotes a child of a joint i, then

$$1 \leq \lambda_j(i) < i < \nu_j(i). \tag{1}$$

This indexing is opposite to (or inverse of) the canonical tree notation used in [33,34]. It follows from (1) that the *root* node (joint) has the number 1, while in [22,23] the root node (link) starts with number 0. For a joint topological tree, a joint has only a single parent but can have multiple children. Let the number of children of the node i be denoted by $\kappa_j(i)$. We call a node that has at most one child a *simple node* and the one that has more than one child a *complex node* (also called junction node in [32]). A simple node that does not have children nodes will be called an *external node*. For example, nodes 3 and 5 in Fig. 1a are complex, nodes 6, 7, 9 and 11 are external, and all remaining nodes are simple. For a root node (joint) $\lambda_j(i) = 0$, and for an external node $\nu_j(i) = 0$. A joint numbering system is not unique, and a given joint topological tree can have multiple correct ways of doing this. Ideally, the knowledge of $\lambda_j(i)$ for each i furnishes complete information about the topological structure of the sys-

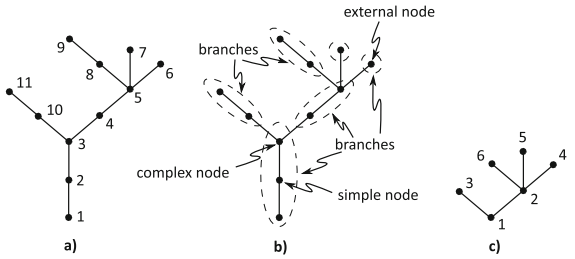

Fig. 1 Topological tree of a mechanical system with 11 joints and 6 branches: **a** joint connectivity graph with node numbers representing joints, **b** branches of the joint connectivity graph, **c** branch connectivity graph with node numbers representing branches

tem which can be expressed, for example, in the form of the so-called *joint parent matrix* $\Upsilon \in \mathbb{R}^{n_j \times 2}$, where n_j is the total number of joints. The first column $\Upsilon(i, 1)$ would specify the current joint number i, and the second column would specify the number of its parent joint $\Upsilon(i, 2) = \lambda_j(i)$. Similarly, complete information is provided if a set of $\nu_j(i)$ is specified for each joint i. Compared to other descriptions of the system topology in terms of *adjacency* and *incidence* matrices[3] [32,34,35,42], the joint parent matrix is already compact. However, the description of the information in the joint parent matrix form can be further compressed and made more efficient.

2.2 Branch connectivity graph and joint-branch connectivity matrix

In this section, we suggest a more efficient (compact) way to describe the topological structure of a mechanical system. Analyzing joint topological trees, we conclude that most nodes (joints) are simple and their description within the joint connectivity graph is rather straightforward, whereas difficulties usually arise in the description of complex nodes. For this purpose, a *branch* of the joint connectivity graph is defined as a set of its nodes that have simple linear connections, Fig. 1b. One might notice that any branch starts either with a root node or with a child of a complex node and ends either with an external node or with a complex node. Definitions of nodes and branches are analogous to those used in electric circuit theory. A *branch connectivity graph* is defined similarly to the joint connectivity graph, but the difference is that each node represents a single branch, Fig. 1c, that is, a node in the former graph represents a single graph branch of the latter one. If a mechanical system has an open-tree structure, its branch connectivity graph is a *branch topological tree*. Similarly to the process of joint numbering, nodes of a branch connectivity graph (open-tree branches of a mechanical system) are numbered so that a given tree branch $i \in \mathbb{N}$ has a lower number than any of its child branches and a higher number than any of its parent branches. For a branch k with a parent $\lambda_b(k)$ and a child $\nu_b(k)$, the identity similar to (1) holds,

$$1 \leq \lambda_b(k) < k < \nu_b(k). \tag{2}$$

For a root node (branch) one has $\lambda_b(k) = 0$ and for an external node $\nu_b(k) = 0$. Simply speaking, a branch connectivity graph is a "lighter" version of a joint connectivity graph where all simple nodes are removed. Therefore, for any node in a branch connectivity graph, one has $\nu_b(k) \neq 1$, which also implies that the branch connectivity graph cannot be further simplified by a similar procedure. In other words, a branch connectivity graph is the simplest representation of the joint connectivity graph which preserves all information about complex nodes. Unless stated otherwise, in what follows the index i refers to the joint number and the index k denotes the branch number. Let $\kappa_b(k)$ denote the number of children of a node in a branch connectivity tree. From the definition of a branch, $\kappa_b(k) = \kappa_j(i)$ where i is the last node of the branch k in the joint connectivity graph, that is, the number of children of the branch k is equal to the number of children of its last joint. In general, $i \neq k$, because the number of joints is not equal to the number of branches. Since our focus is on a branch connectivity graph and not on simple joint nodes, for the sake of simplicity, we slightly abuse the notation denoting by $\kappa(i)$ the number of children of a node. Complete information about the branch topological tree of a given system is provided if for each branch either $\lambda_b(k)$ or $\nu_b(k)$ is specified. This can be achieved by specifying a *branch parent matrix* $\Phi \in \mathbb{R}^{n_b \times 2}$, where n_b is the total number of branches in the branch topological tree for the given system. The first column of the matrix shows that the current branch number $\Phi(k, 1) = k$, and the second column specifies its parent branch number $\Phi(k, 2) = \lambda_b(k)$. The branch numbering is not unique and can be performed in multiple ways.

[3] The adjacency matrix is a square matrix with a dimension equal to the number of nodes, while the number of rows and columns in the incidence matrix is equal to the number of nodes and arcs, respectively.

When the branch connectivity graph is specified, one cannot construct the kinematic tree of the dynamic system yet because the knowledge about the joints that compose each branch is missing. This information can be provided in a *joint-branch connectivity matrix* $\Psi \in \mathbb{R}^{n_b \times 2}$. Under the assumption that the joints in a given branch are numbered consecutively, the first column in $\Psi(k, 1)$ specifies the joint number where the branch k starts, whereas the second column $\Psi(k, 2)$ specifies the joint number where it terminates. There are two alternative ways to describe system's kinematics structure, using either the joint connectivity graph (this requires a single matrix Υ) or the branch connectivity graph in conjunction with the joint-branch connectivity matrix (this requires two matrices Φ and Ψ). If a mechanical system possesses many joints and a few branches, using the latter approach is simpler and more efficient whereas for a system with many branches and joints the former description is preferable. For example, for a system with one hundred DOF and only four branches, the branch topological tree has four nodes, and the joint-branch connectivity matrix is a 4×2 matrix. Recall that a node represents a joint in a joint connectivity graph and a branch in a branch connectivity graph. Therefore, for a system whose kinematic structure is represented by a full binary tree, the joint connectivity graph requires less information. Our method to describe the system's topology is also different (simpler) than in [33]. However, this simplicity is only related to the description of the system's topology, and it does not affect the sensitivity analysis. In other words, the sensitivity analysis does not depend much on the method to describe the topology of the system.

2.3 Denavit–Hartenberg convention

The Denavit-Hartenberg (D-H) convention is used to formulate the kinematics of the model. It specifies four parameters for each DOF, θ_i, d_i, a_i, and α_i, where the variables θ_i and d_i are associated with the cases where the joint is revolute or prismatic, respectively. We note that there are multiple variations of the D-H convention. In this paper, the distal version of the D-H convention is utilized. This means that the i-th local frame is attached to the distal end of the link i [24], and a joint i is located at the proximal end of the same link. Alternatively, the same local frame can be attached to the proximal end of the link [12]. Even though all D-H conventions describe the kinematics of arbitrary mechanical systems equally well, care should be taken when working with the D-H convention because of the aforementioned notational differences.

3 Dynamics

Assume that we have a mechanical n_q-DOF system, then the number of joints $n_q = n_j$. For a *fully actuated* system, the number of input (driving) torques is equal to the number of DOF. If the number of input torques is less than the number of joints, then one has an *under-actuated system*; otherwise the system is *over-actuated*. A human body is an over-actuated system, which is a beneficial feature in terms of redundancy, because a single muscle can actuate multiple joints and a single joint can be actuated by multiple muscles. However, this redundancy is difficult to model, especially when our goal is to generate a dynamically feasible human body motion. To achieve this, it suffices to consider a fully actuated human body model which determines our main focus on fully actuated multibody systems. The detailed derivation of the dynamic equations of motion is outside the scope of this paper. Our work is based on the Lagrangian formulation, so a reader interested in more details about the derivation (such as Lagrangian function), can consult [1]. Dynamic equations of motion for a mechanical n_q-DOF system are often represented in the form

$$M(q)\ddot{q} + K(q, \dot{q}) + W(q) = Q, \tag{3}$$

where $q = [q_1 \; q_2 \; \dots q_{n_q}]^T \in \mathbb{R}^{n_q}$ and $\dot{q} = [\dot{q}_1 \; \dot{q}_2 \; \dots \; \dot{q}_{n_q}]^T \in \mathbb{R}^{n_q}$ are the vectors of generalized coordinates and generalized velocities, respectively, $M(q) \in \mathbb{R}^{n_q \times n_q}$ is the inertia matrix, $K(q, \dot{q}) \in \mathbb{R}^{n_q}$ is the vector of Coriolis and normal inertial forces, $W(q) \in \mathbb{R}^{n_q}$ is the vector of gravitational forces, and $Q = [Q_1 \; Q_2 \; \dots \; Q_{n_q}]^T \in \mathbb{R}^{n_q}$ is the vector of generalized forces. The superscript $(\cdot)^T$ denotes the transpose operator. These equations can be referred to as "bulk" form, because all DOF are included in the equation, and are therefore computed simultaneously [24]. However, the direct computation of this set of equations is burdensome, especially for high-DOF systems with many joints and links. Instead, there are efficient "recursive" techniques of calculation where each DOF is computed recursively from the previous DOF [11,30,41]. We use the recursive Lagrangian formulation to derive the dynamic equations of motion based on the extension of ideas reported in [1,57,58] to the case of multiple external loads and topological tree structures.

3.1 Recursive kinematics

For each DOF, the corresponding homogeneous transformation matrix $A_i \in \mathbb{R}^{4\times4}$ is defined as

$$A_i=\begin{bmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \cos\theta_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1. \end{bmatrix}, \tag{4}$$

where $i = 1, \ldots, n_q$ [1]. The matrix transforms homogeneous coordinates from the frame i to its parent frame $i-1$. The first, second and third-order derivatives of A_i with respect to q_i are defined accordingly as $B_i = dA_i/dq_i$, $C_i = dB_i/dq_i$ and $D_i = dC_i/dq_i$. The matrices A_i, B_i, C_i, and D_i are related by the equations $B_i = ZA_i$, $C_i = ZB_i$, $D_i = ZC_i$ [24] where the matrix $Z \in \mathbb{R}^{4\times4}$ has all zero entries except for the following: $Z(1,2) = -1$ and $Z(2,1) = 1$ for a revolute joint, and $Z(3,4) = 1$ for a prismatic joint. The global homogeneous transformation matrix is introduced using the local transformation matrices as $T_i = A_1 \cdot \ldots \cdot A_i \in \mathbb{R}^{4\times4}$. The first and second-order time derivatives of the global transformation matrix are $S_i = dT_i/dt = \dot{T}_i \in \mathbb{R}^{4\times4}$ and $R_i = dS_i/dt = \dot{S}_i \in \mathbb{R}^{4\times4}$. These matrices can be calculated in recursive form starting from the first, root joint, until the last, end-effector joint, in the kinematic chain:

$$\begin{aligned} T_i &= T_p A_i, \\ S_i &= S_p A_i + T_p B_i \dot{q}_i, \quad i = 1, \ldots, n_q \\ R_i &= R_p A_i + 2S_p B_i \dot{q}_i + T_p C_i \dot{q}_i^2 + T_p B_i \ddot{q}_i. \end{aligned} \tag{5}$$

where $\ddot{q}_i = d\dot{q}_i/dt$. For a system which has only simple nodes in the joint topological tree or for a single branch within the joint topological tree, the subscript $p = i-1$. For a node that has a complex node parent, if the joint connectivity graph is used, the subscript $p = \lambda_j(i) = \Upsilon(i,2)$ and if the branch connectivity graph is used, $p = \Psi(\lambda_b(k),2) = \Psi(\Phi(k,2),2)$, where k is the index of the branch to which the joint i belongs. For the first joint ($i = 1$), one sets $p = 0$. In summary, if $i = \Psi(k,1) \wedge \lambda_b(k) \neq 0$, then $p = \Psi(\Phi(k,2),2)$, if $i = \Psi(k,1) \wedge \lambda_b(k) = 0$, then $p = 0$, otherwise, $p = i-1$. The initial values of the global transformation matrix T_i and its time derivatives are $T_0 = I$, $S_0 = 0$, and $R_0 = 0$, where $I \in \mathbb{R}^{4\times4}$ and $0 \in \mathbb{R}^{4\times4}$ are, correspondingly, the identity and zero matrices. With the help of homogeneous matrices, the position r_g, velocity v_g, and acceleration a_g of any point of the mechanical system with respect to the global frame can be specified as

$$r_g = T_i r_i, \quad v_g = S_i r_i, \quad a_g = R_i r_i, \tag{6}$$

where r_i is the position of the point in the local coordinate system i. The set of equations in (5) defines what is called the *recursive forward kinematics*.

3.2 Recursive dynamics

After the forward kinematics matrices are computed, dynamic matrices can be obtained. Firstly, we define the mass $m_i \in \mathbb{R}$ and inertia matrix $J_i \in \mathbb{R}^{4\times4}$ of the link associated with a joint i and expressed in the local frame i. The inertia matrix is defined as

$$J_i = \begin{bmatrix} J^*_{xx} & -J_{xy} & -J_{xz} & m_i\, {}^c x_i \\ -J_{xy} & J^*_{yy} & -J_{yz} & m_i\, {}^c y_i \\ -J_{xz} & -J_{yz} & J^*_{zz} & m_i\, {}^c z_i \\ m_i\, {}^c x_i & m_i\, {}^c y_i & m_i\, {}^c z_i & m_i. \end{bmatrix}, \tag{7}$$

where $J^*_{xx} = \frac{1}{2}(-J_{xx}+J_{yy}+J_{zz})$, $J^*_{yy} = \frac{1}{2}(J_{xx}-J_{yy}+J_{zz})$, $J^*_{zz} = \frac{1}{2}(J_{xx}+J_{yy}-J_{zz})$. J_{ij}, $i, j \in x, y, z$, are the moments and products of inertia of link i in its local frame of reference, while ${}^c x_i$, ${}^c y_i$, and ${}^c z_i$ denote the location of the center of mass of link i in the coordinate system i. Secondly, the inertia $H_i \in \mathbb{R}^{4\times4}$ and the external force $F_i \in \mathbb{R}^{4\times n_q}$ matrices, the gravity $E_i \in \mathbb{R}^4$ and the external torque $G_i \in \mathbb{R}^4$ vectors are defined. For simplicity, we refer to these matrices as *dynamic transformation matrices*, because they transform dynamic quantities from a given joint to the next one. These matrices are also defined recursively, but this time from the last joint (end-effector) to the first joint (root)

$$\begin{aligned} H_i &= J_i R_i^T + \sum_{p=1}^{\kappa(i)} A_p H_p, \\ E_i &= m_i\, {}^c r_i + \sum_{p=1}^{\kappa(i)} A_p E_p, \quad i = 1, \ldots, n_q, \\ F_i &= {}^f R_i + \sum_{p=1}^{\kappa(i)} A_p F_p, \; G_i = h_i + \sum_{p=1}^{\kappa(i)} G_p. \end{aligned} \tag{8}$$

If a system does not have complex nodes in its joint topological tree or within a branch $\kappa(i) = 1$, the subscript $p = i+1$. For a complex node in the joint topological tree $p = \nu_j(i)$, and in the branch topological tree $p = \Psi(\nu_b(k),1) = \Psi(\Phi(j(k),1),1)$,

where, as above, k is the index of the branch to which the joint i belongs. The index notation $j(k)$ is used for the children of the branch k found through the branch parent matrix by using the identity $k = \Phi(j, 2)$. For the last joint in the branch without child joints, $\kappa(i) = 0$; we can also assume that the initial values of the dynamic matrices and vectors are $D_{n_q+1} = 0$ and $E_{n_q+1} = F_{n_q+1} = G_{n_q+1} = 0$. The same initial values are used for any external node in the joint connectivity graph. In summary, if $i = \Psi(k, 2) \wedge \nu_b(k) \neq 0$, then $p = \Psi(\Phi(j(k), 1), 1)$, if $i = \Psi(k, 2) \wedge \nu_b(k) = 0$, then $\kappa(i) = 0$, and if $i \neq \Psi(k, 2)$, then $p = i + 1$ and $\kappa(i) = 1$. Furthermore, ${}^c r_i = [{}^c x_i \ {}^c y_i \ {}^c z_i \ 1]^T$ is the location of the center of mass of link i with respect to frame i, where the matrix ${}^f R_i \in \mathbb{R}^{4\times n_q}$ has all zero entries except for those in the only non-zero column i, ${}^f R_i(:, i) = {}^f r_i = [{}^f x_i \ {}^f y_i \ {}^f z_i \ 1]^T$. This is the location of the point of application of an external point force f_i acting on the link i expressed in the local frame i. The net external force $f_i = [f_{i,x} \ f_{i,y} \ f_{i,z} \ 0]^T$ and torque $h_i = [h_{i,x} \ h_{i,y} \ h_{i,z} \ 0]^T$ acting on the link i are expressed in the global frame.[4] The net external forces and torques acting on all links can be grouped into matrices $V = [f_1 \ f_2 \cdots f_{n_q}] \in \mathbb{R}^{4\times n_q}$ and $U = [h_1 \ h_2 \cdots h_{n_q}] \in \mathbb{R}^{4\times n_q}$.

The generalized force Q_i acting on the link i (i-th component of the vector Q from (3)) can be found by a similar recursive process starting from the last link to the first link,

$$Q_i = \operatorname{tr}\left[\frac{\partial T_i}{\partial q_i} H_i\right] - g^T \frac{\partial T_i}{\partial q_i} E_i - \operatorname{tr}\left[V^T \frac{\partial T_i}{\partial q_i} F_i\right]_{i:n_q} - G_i^T T_p w_0, \tag{9}$$

for $i = 1, \ldots, n_q$, $\operatorname{tr}[\cdot]$ is the trace operator, and $\operatorname{tr}[\cdot]_{i:n_q}$ denotes the sum of diagonal matrix elements from i until n_q, $g = [g_x \ g_y \ g_z \ 0]^T$ is the gravity vector expressed in the global frame of reference,[5] the vector w_0 is defined as $w_0 = [0\,0\,1\,0]^T$ for a revolute joint and as $w_0 = 0 \in \mathbb{R}^4$ for a prismatic joint. The subscript p in (9) is defined as in (5): if $i = \Psi(k, 1) \wedge \lambda_b(k) \neq 0$, then $p = \Psi(\Phi(k, 2), 2)$, if $i = \Psi(k, 1) \wedge \lambda_b(k) = 0$, then $p = 0$, otherwise $p = i - 1$. The external force f_i and the torque h_i are both expressed in the global coordinate system. The process of calculating the torque and the terms defined in (8) is called the *recursive backward dynamics*.

[4] If there are multiple contact and/or distributed forces and torques acting on the link, then we can always find their net resultants.

[5] in our case $g = [0\,0\ -9.81\,0]^T$.

4 Sensitivity analysis

In gradient-based optimization algorithms, convergence to the local minimum is highly dependent on the gradient information. It is known that the computation of the derivatives of the objective function and constraints in some cases consumes over 90% of the CPU time required for each iteration of the algorithm [2]. The computation time depends on the selection of the set of generalized coordinates, constraints, constraints enforcement schemes, the set of sensitivity parameters, differentiation methods, and other factors. Thus, for solving an NLP problem efficiently, it is desirable to provide gradients of the object function and constrains with respect to optimization variables. Furthermore, the gradient and Jacobian information enable a coherent and easy extension of algorithms to more complex models making algorithms well scalable. To this end, a sensitivity analysis is performed, i.e., partial derivatives of the desired expressions with respect to the generalized coordinates, velocities, and accelerations are determined. There are two reasons for the utilization of the direct differentiation method for sensitivity analysis in this paper. The first is that the adjoint variable method requires integration of the adjoint sensitivity equations. However, we use the inverse dynamics formulation to avoid the integration of dynamic equations of motion. Thus, the adjoint variable method would neutralize our effort to avoid the time integration. Secondly, as already mentioned, our work is extension of the work in [1], and so we continue to use the direct differentiation methods as they did.

4.1 Kinematic sensitivity analysis

The sensitivity of homogeneous transformation matrices with respect to the generalized coordinates, velocities and accelerations is written as

$$\frac{\partial T_i}{\partial q_m}, \ \frac{\partial S_i}{\partial q_m}, \ \frac{\partial S_i}{\partial \dot{q}_m}, \ \frac{\partial R_i}{\partial q_m}, \ \frac{\partial R_i}{\partial \dot{q}_m}, \ \frac{\partial R_i}{\partial \ddot{q}_m}, \ \frac{\partial^2 T_i}{\partial q_m \partial q_i} \in \mathbb{R}^{4\times 4},$$

where $i = 1, \ldots, n_q, m = 1, \ldots, n_q$. Since T_i depends only on the generalized coordinates, its partial derivatives with respect to generalized velocities and accelerations are zeros. Similarly, partial derivatives of S_i with respect to generalized accelerations are zeros. The

homogeneous transformation matrices can be found from the following relations. For $m < i$,

$$\begin{aligned}
\frac{\partial T_i}{\partial q_m} &= \frac{\partial T_p}{\partial q_m} A_i, \\
\frac{\partial S_i}{\partial q_m} &= \frac{\partial S_p}{\partial q_m} A_i + \frac{\partial T_p}{\partial q_m} B_i \dot{q}_i, \\
\frac{\partial S_i}{\partial \dot{q}_m} &= \frac{\partial S_p}{\partial \dot{q}_m} A_i, \\
\frac{\partial R_i}{\partial q_m} &= \frac{\partial R_p}{\partial q_m} A_i + 2\frac{\partial S_p}{\partial q_m} B_i \dot{q}_i + \frac{\partial T_p}{\partial q_m} C_i \dot{q}_i^2 + \frac{\partial T_p}{\partial q_m} B_i \ddot{q}_i, \\
\frac{\partial R_i}{\partial \dot{q}_m} &= \frac{\partial R_p}{\partial \dot{q}_m} A_i + 2\frac{\partial S_p}{\partial \dot{q}_m} B_i \dot{q}_i, \\
\frac{\partial R_i}{\partial \ddot{q}_m} &= \frac{\partial R_p}{\partial \ddot{q}_m} A_i, \\
\frac{\partial^2 T_i}{\partial q_m \partial q_i} &= \frac{\partial T_p}{\partial q_m} B_i.
\end{aligned} \tag{10}$$

For the case $(m = i)$:

$$\begin{aligned}
\frac{\partial T_i}{\partial q_m} &= T_p B_i, \\
\frac{\partial S_i}{\partial q_m} &= S_p B_i + T_p C_i \dot{q}_i, \\
\frac{\partial S_i}{\partial \dot{q}_m} &= T_p B_i, \\
\frac{\partial R_i}{\partial q_m} &= R_p B_i + 2 S_p C_i \dot{q}_i + T_p D_i \dot{q}_i^2 + T_p C_i \ddot{q}_i, \\
\frac{\partial R_i}{\partial \dot{q}_m} &= 2 S_p B_i + 2 T_p C_i \dot{q}_i, \\
\frac{\partial R_i}{\partial \ddot{q}_m} &= T_p B_i, \\
\frac{\partial^2 T_i}{\partial q_m \partial q_i} &= T_p C_i.
\end{aligned} \tag{11}$$

For the case $(m > i) \vee (m < i \wedge i = \Psi(k, 1) \wedge \lambda_b(k) = 0)$:

$$\begin{aligned}
&\frac{\partial T_i}{\partial q_m} = \frac{\partial S_i}{\partial q_m} = \frac{\partial S_i}{\partial \dot{q}_m} = \frac{\partial R_i}{\partial q_m} \\
&= \frac{\partial R_i}{\partial \dot{q}_m} = \frac{\partial R_i}{\partial \ddot{q}_m} = \frac{\partial^2 T_i}{\partial q_m \partial q_i} = 0
\end{aligned} \tag{12}$$

Note that $p = \Psi(\lambda_b(k), 2) = \Psi(\Phi(k, 2), 2)$ in (10) and (11) if, in addition to the conditions imposed at the beginning of each equation, $i = \Psi(k, 1) \wedge \lambda_b(k) \neq 0$. These additional conditions are used to determine whether a parent branch exists. The index k refers to the branch number associated with the joint number i, e.g., it is understood that the joint i is located within the branch k. If $i = \Psi(k, 1) \wedge \lambda_b(k) = 0$, then $p = 0$ in (11) for $m = i$, but for $m < i$ (12) applies. If $i \neq \Psi(k, 1)$, then $p = i - 1$. Alternatively, in the joint connectivity graph notation, $p = \lambda_j(i) = \Upsilon(i, 2)$.

4.2 Dynamic sensitivity analysis

Similarly to the kinematics sensitivity analysis, sensitivity of the dynamic transformation matrices with respect to generalized coordinates, velocities and accelerations is written as

$$\frac{\partial H_i}{\partial q_m}, \frac{\partial H_i}{\partial \dot{q}_m}, \frac{\partial H_i}{\partial \ddot{q}_m} \in \mathbb{R}^{4\times 4}, \frac{\partial E_i}{\partial q_m}, \frac{\partial G_i}{\partial q_m} \in \mathbb{R}^{4\times 1}, \frac{\partial F_i}{\partial q_m} \in \mathbb{R}^{4\times n_q}$$

where $i = 1, \ldots, n_q$, $m = 1, \ldots, n_q$. These matrices can be found from the following relations. For $m \leq i$,

$$\begin{aligned}
\frac{\partial H_i}{\partial q_m} &= J_i \frac{\partial R_i^T}{\partial q_m} + \sum_{p=1}^{\kappa(i)} A_p \frac{\partial H_p}{\partial q_m}, \\
\frac{\partial H_i}{\partial \dot{q}_m} &= J_i \frac{\partial R_i^T}{\partial \dot{q}_m} + \sum_{p=1}^{\kappa(i)} A_p \frac{\partial H_p}{\partial \dot{q}_m}, \\
\frac{\partial H_i}{\partial \ddot{q}_m} &= J_i \frac{\partial R_i^T}{\partial \ddot{q}_m} + \sum_{p=1}^{\kappa(i)} A_p \frac{\partial H_p}{\partial \ddot{q}_m}, \\
\frac{\partial E_i}{\partial q_m} &= 0, \\
\frac{\partial F_i}{\partial q_m} &= 0, \\
\frac{\partial G_i}{\partial q_m} &= 0.
\end{aligned} \tag{13}$$

For $m \geq i + 1$,

$$\begin{aligned}
\frac{\partial H_i}{\partial q_m} &= \sum_{p=1}^{\kappa(i)} A_p \frac{\partial H_p}{\partial q_m} + \beta B_p H_p, \\
\frac{\partial H_i}{\partial \dot{q}_m} &= \sum_{p=1}^{\kappa(i)} A_p \frac{\partial H_p}{\partial \dot{q}_m}, \\
\frac{\partial H_i}{\partial \ddot{q}_m} &= \sum_{p=1}^{\kappa(i)} A_p \frac{\partial H_p}{\partial \ddot{q}_m}, \\
\frac{\partial E_i}{\partial q_m} &= \sum_{p=1}^{\kappa(i)} A_p \frac{\partial E_p}{\partial q_m} + \beta B_p E_p, \\
\frac{\partial F_i}{\partial q_m} &= \sum_{p=1}^{\kappa(i)} A_p \frac{\partial F_p}{\partial q_m} + \beta B_p F_p, \\
\frac{\partial G_i}{\partial q_m} &= 0.
\end{aligned} \tag{14}$$

For the case $m \geq i + 1 \wedge \Psi(k, 2) \wedge \nu_b(k) = 0$,

$$\frac{\partial H_i}{\partial q_m} = \frac{\partial H_i}{\partial \dot{q}_m} = \frac{\partial H_i}{\partial \ddot{q}_m}$$

$$= \frac{\partial E_i}{\partial q_m} = \frac{\partial F_i}{\partial q_m} = \frac{\partial G_i}{\partial q_m} = 0 \tag{15}$$

where $p = \Psi(\nu_b(k), 1) = \Psi(\Phi(j(k), 1), 1)$ provided that, in addition to the conditions imposed at the beginning of each equation, $i = \Psi(k, 2) \wedge \nu_b(k) \neq 0$. These conditions verify the existence of children of the branch k and the sum includes all its child branches $\kappa(i)$. If $i = \Psi(k, 2) \wedge \nu_b(k) = 0$, then $\kappa(i) = 0$ in (13) for $m \leq i$, but for $m \geq i + 1$ (15) applies. In the case $i \neq \Psi(k, 2)$, one has $p = i + 1$ and $\kappa(i) = 1$. The coefficient $\beta = 1$ if $m = p$, which means that $m = \Psi(\nu_b(k), 1)$ for the case $i = \Psi(k, 2)$ and $m = i + 1$ for the case $i \neq \Psi(k, 2)$, otherwise, $\beta = 0$. In the joint connectivity graph notation, the index $p = \nu_j(i)$.

Finally, the torque sensitivity can be found as follows.

$$\begin{aligned}
\frac{\partial Q_i}{\partial q_m} &= \mathrm{tr}\left[\frac{\partial^2 T_i}{\partial q_m \partial q_i} H_i + \frac{\partial T_i}{\partial q_i}\frac{\partial H_i}{\partial q_m}\right] - g^T \frac{\partial^2 T_i}{\partial q_m \partial q_i} E_i \\
&- \mathrm{tr}\left[V^T \frac{\partial^2 T_i}{\partial q_m \partial q_i} F_i\right]_{i:n_q} - G_i^T \frac{\partial T_p}{\partial q_m} w_0, \qquad m \leq i, \\
\frac{\partial Q_i}{\partial q_m} &= \mathrm{tr}\left[\frac{\partial T_i}{\partial q_i}\frac{\partial H_i}{\partial q_m}\right] - g^T \frac{\partial T_i}{\partial q_i}\frac{\partial E_i}{\partial q_m} - \mathrm{tr}\left[V^T \frac{\partial T_i}{\partial q_i}\frac{\partial F_i}{\partial q_m}\right]_{i:n_q}, \\
& m \geq i + 1,
\end{aligned} \tag{16}$$

$$\begin{aligned}
\frac{\partial Q_i}{\partial \dot{q}_m} &= \mathrm{tr}\left[\frac{\partial T_i}{\partial q_i}\frac{\partial H_i}{\partial \dot{q}_m}\right], \\
\frac{\partial Q_i}{\partial \ddot{q}_m} &= \mathrm{tr}\left[\frac{\partial T_i}{\partial q_i}\frac{\partial H_i}{\partial \ddot{q}_m}\right]
\end{aligned} \tag{17}$$

where, as in (9), $p = \Psi(\Phi(k, 2), 2)$ if $i = \Psi(k, 1) \wedge \lambda_b(k) \neq 0$ and $p = 0$ if $i = \Psi(k, 1) \wedge \lambda_b(k) = 0$, otherwise, $p = i - 1$. Equations (10)-(17) define a recursive kinematics and dynamic sensitivity analysis. The corresponding pseudo-algorithms are summarized in Algorithms 1 -3, which require three loops, the kinematic loop from the root to the external nodes, the dynamic loop from external nodes to the root, and, finally, the torque loop back again from the root to external nodes. It is possible to include computations in Algorithm 3 in Algorithm 2 so that only two loops are required. For the sake of clarity, however, we present the sensitivity analysis in three loops.

5 Simulation

5.1 Human model

A rigid human body model consisting of rigid links and revolute joints is used to test the proposed sensitivity method. A complete human body has about 350 joints [40]. For our purposes, the human model has 43 DOF and 20 links. The number of joints exceeds the number of links due to the presence of virtual links which have zero length and mass. For example, a spherical joint is represented as a combination of three consecutive rotational joints. The 43-DOF human model is shown schematically in Fig. 2 where the green dot indicates the origin of the global coordinate system. The red dot indicates the *pelvis* point used as the reference point for the human body model. The global coordinate system has axes x_0-y_0-z_0, and the local coordinate system i, associated with the joint i and attached to the distal end of the link i, has the unit vectors x_i-y_i-z_i. The mobility of the human model with respect to the global frame of reference is achieved by the global virtual DOF of the joints connected to the pelvis point. Global virtual DOF consists of three translational and three rotational DOF. All other human links and joints are referenced through the pelvis point. The joint and branch connectivity graphs for the human model are shown in Fig. 3. There are 43 joints and 7 branches. The corresponding joint parent matrix $\Upsilon \in \mathbb{R}^{43\times 2}$, the branch parent matrix $\Phi \in \mathbb{R}^{7\times 2}$, and the joint-branch connectivity matrix $\Psi \in \mathbb{R}^{7\times 2}$ are defined by (18)-(20). Even for this modestly-sized system, the branch connectivity graph allows a more compact and efficient description of the system topology in comparison with the joint connectivity graph. The values of D-H parameters for the human model are collected in Table 1. The values of the physical parameters for the 43 DOF human model are provided in Table 2. Note that the *neutral position* of

Algorithm 1 Forward Recursive Kinematic and Sensitivity Analysis

1: **for** $k_I = 1, \ldots, n_b$ **do**
2: **for** $i = \Psi(k_I, 1), \ldots, \Psi(k_I, 2)$ **do**
3: For a given time t, obtain the generalized coordinates, velocities, and accelerations q, $\dot{q}$ and $\ddot{q}$.
4: Evaluate local homogeneous transformation matrices A_i, B_i, C_i, and D_i.
5: Evaluate recursively the global homogeneous transformation matrices T_i, S_i, and R_i using (5).
6: **for** $k_{II} = 1, \ldots, n_b$ **do**
7: **for** $m = \Psi(k_{II}, 1), \ldots, \Psi(k_{II}, 2)$ **do**
8: Evaluate kinematic sensitivity matrices $\frac{\partial T_i}{\partial q_m}$, $\frac{\partial S_i}{\partial q_m}$, $\frac{\partial S_i}{\partial \dot{q}_m}$, $\frac{\partial R_i}{\partial q_m}$, $\frac{\partial R_i}{\partial \dot{q}_m}$, $\frac{\partial R_i}{\partial \ddot{q}_m}$, $\frac{\partial^2 T_i}{\partial q_m \partial q_i}$ using (10)-(12).
9: **end for**
10: **end for**
11: **end for**
12: **end for**

Algorithm 2 Backward Recursive Dynamic and Sensitivity Analysis

for $k_I = n_b, \ldots, 1$ **do**
  for $i = \Psi(k_I, 2), \ldots, \Psi(k_I, 1)$ **do**
    For a given time t, obtain the generalized coordinates, velocities, and accelerations q, $\dot{q}$ and $\ddot{q}$.
    Evaluate recursively the dynamic transformation matrices H_i, E_i, F_i, and G_i using (8).
    for $k_{II} = n_b, \ldots, 1$ **do**
      for $m = \Psi(k_{II}, 2), \ldots, \Psi(k_{II}, 1)$ **do**
        Evaluate dynamic sensitivity matrices $\frac{\partial H_i}{\partial q_m}$, $\frac{\partial H_i}{\partial \dot{q}_m}$, $\frac{\partial H_i}{\partial \ddot{q}_m}$, $\frac{\partial E_i}{\partial q_m}$, $\frac{\partial F_i}{\partial q_m}$, $\frac{\partial G_i}{\partial q_m}$ using (13)-(15).
      end for
    end for
  end for
end for

Algorithm 3 Forward Recursive Torque and its Sensitivity Analysis

for $k_I = 1, \ldots, n_b$ **do**
  for $i = \Psi(k_I, 1), \ldots, \Psi(k_I, 2)$ **do**
    Evaluate torque Q_i values from (9).
    for $k_{II} = 1, \ldots, n_b$ **do**
      for $m = \Psi(k_{II}, 1), \ldots, \Psi(k_{II}, 2)$ **do**
        Evaluate torque sensitivity matrices $\frac{\partial Q_i}{\partial q_m}$, $\frac{\partial Q_i}{\partial \dot{q}_m}$, $\frac{\partial Q_i}{\partial \ddot{q}_m}$ using (16)-(17).
      end for
    end for
  end for
end for

the human model corresponds to the vertical rest position (the human model stands vertically straight with both arms stretched along the sides, as shown in Fig. 2).

$$\Upsilon^T = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 5 & 14 & 15 & 16 & 17 & 18 \\ 20 & 21 & 22 & 23 & 24 & 25 & 26 & 27 & 28 & 29 & 30 & 31 & 32 & 33 & 34 & 35 & & & \\ 19 & 20 & 5 & 22 & 23 & 24 & 25 & 26 & 27 & 28 & 29 & 30 & 22 & 32 & 33 & 34 & & & \\ 35 & 36 & 37 & 38 & 39 & 40 & 41 & 42 & 43 & & & & & & & & & & \\ 34 & 35 & 36 & 37 & 38 & 39 & 22 & 41 & 42 & & & & & & & & & & \end{bmatrix}. \tag{18}$$

$$\Phi^T = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 0 & 1 & 1 & 1 & 4 & 4 & 4 \end{bmatrix}. \tag{19}$$

$$\Psi^T = \begin{bmatrix} 1 & 6 & 14 & 22 & 23 & 32 & 41 \\ 5 & 13 & 21 & 22 & 31 & 40 & 43 \end{bmatrix}. \tag{20}$$

5.2 Implementation

We performed the sensitivity analysis using the MATLAB software. Due to the inverse dynamics formulation utilized in this work, the reference trajectories

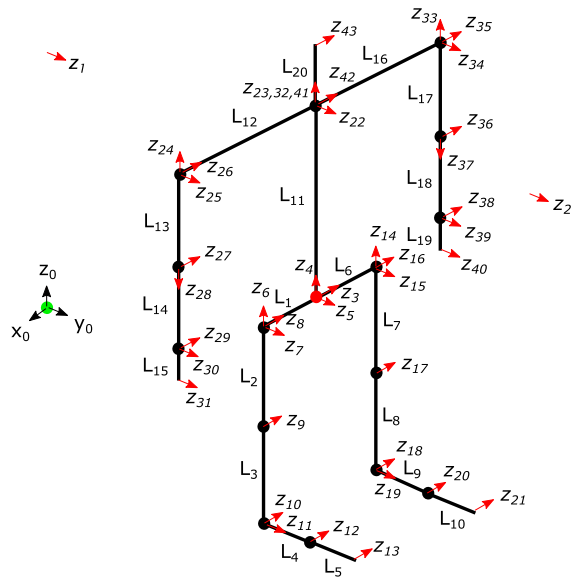

Fig. 2 Schematic diagram of the humanoid model with 43 DOF and ten links. For clarity, only the z-axes of the local coordinate systems are shown

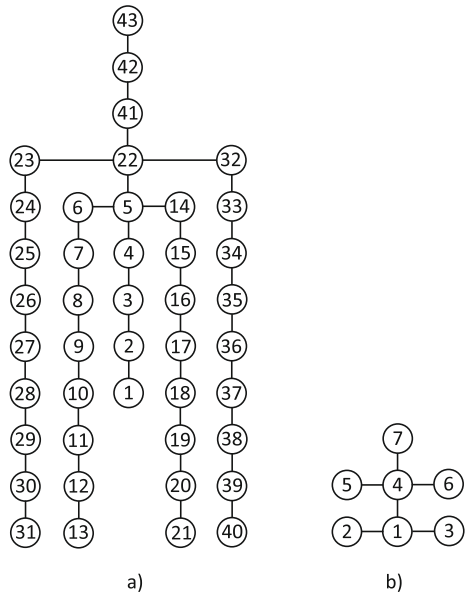

Fig. 3 Joint (**a**) and branch (**b**) connectivity graphs for the human model

for generalized coordinates, velocities, and accelerations are required for the sensitivity analysis. As reference trajectories for the joints, we simply imposed the sinusoidal function $q_i = \sin 2\pi f t$, where $f = 1$ and t are frequency and time, respectively. The generalized velocities and accelerations can be obtained by

Table 1 D-H table for the 43 DOF humanoid model. Index i refers to the joint (DOF) number, while index k refers to the branch number

D-H Table					
i	k	θ_i	d_i	a_i	α_i
1	1	0	q_1	0	$-\frac{\pi}{2}$
2	1	$\frac{\pi}{2}$	q_2	0	$-\frac{\pi}{2}$
3	1	0	$-q_3$	0	0
4	1	$q_4+\frac{\pi}{2}$	0	0	$-\frac{\pi}{2}$
5	1	$q_5+\frac{\pi}{2}$	0	0	$-\frac{\pi}{2}$
6	2	q_6	0	L_1	$\frac{\pi}{2}$
7	2	q_7	0	0	$-\frac{\pi}{2}$
8	2	$q_8+\frac{\pi}{2}$	0	0	$-\frac{\pi}{2}$
9	2	q_9	0	L_2	0
10	2	q_{10}	0	L_3	0
11	2	q_{11}	0	0	$\frac{\pi}{2}$
12	2	q_{12}	L_4	0	$-\frac{\pi}{2}$
13	2	$q_{13}-\frac{\pi}{2}$	0	L_5	0
14	3	q_{14}	0	$-L_6$	$\frac{\pi}{2}$
15	3	q_{15}	0	0	$-\frac{\pi}{2}$
16	3	$q_{16}+\frac{\pi}{2}$	0	0	$-\frac{\pi}{2}$
17	3	q_{17}	0	L_7	0
18	3	q_{18}	0	L_8	0
19	3	q_{19}	0	0	$\frac{\pi}{2}$
20	3	q_{20}	L_9	0	$-\frac{\pi}{2}$
21	3	$q_{21}-\frac{\pi}{2}$	0	L_{10}	0
22	4	$q_{22}+\frac{\pi}{2}$	0	$-L_{11}$	0
23	5	$q_{23}-\frac{\pi}{2}$	0	0	$\frac{\pi}{2}$
24	5	q_{24}	0	L_{12}	0
25	5	q_{25}	0	0	$-\frac{\pi}{2}$
26	5	$q_{26}+\frac{\pi}{2}$	0	0	$-\frac{\pi}{2}$
27	5	q_{27}	0	L_{13}	0
28	5	$q_{28}-\frac{\pi}{2}$	0	0	$-\frac{\pi}{2}$
29	5	q_{29}	L_{14}	0	$\frac{\pi}{2}$
30	5	$q_{30}+\frac{\pi}{2}$	0	0	$\frac{\pi}{2}$
31	5	q_{31}	0	L_{15}	0
32	6	$q_{32}-\frac{\pi}{2}$	0	0	$\frac{\pi}{2}$
33	6	q_{33}	0	$-L_{16}$	0
34	6	q_{34}	0	0	$-\frac{\pi}{2}$
35	6	$q_{35}+\frac{\pi}{2}$	0	0	$-\frac{\pi}{2}$
36	6	q_{36}	0	L_{17}	0
37	6	$q_{37}-\frac{\pi}{2}$	0	0	$-\frac{\pi}{2}$
38	6	q_{38}	L_{18}	0	$\frac{\pi}{2}$
39	6	$q_{39}+\frac{\pi}{2}$	0	0	$\frac{\pi}{2}$
40	6	q_{40}	0	L_{19}	0
41	7	$q_{41}-\frac{\pi}{2}$	0	0	$\frac{\pi}{2}$
42	7	$q_{42}+\frac{\pi}{2}$	0	0	$-\frac{\pi}{2}$
43	7	$q_{43}+\frac{\pi}{2}$	0	$-L_{20}$	0

a simple time differentiation of the generalized coordinates. These reference trajectories were discretized with a sampling time of 10 ms, whereas the duration of the simulation was set to 3 s, resulting in a total of 301 sampling instances. At each sampling instance, both numerical and suggested analytical sensitivity values were evaluated. Numerical sensitivity was estimated from (9) using the `jacobianest` function of MATLAB, which is a fully adaptive and robust numerical differentiation tool [14]. The analytical sensitivity was obtained from (16)-(17). Note that for a system with 43 DOF, the total number of different torque sensitivity values $\frac{\partial Q_i}{\partial q_m}$, $i, m = 1, \ldots, 43$, is equal to $43^2 = 1849$. Similarly for the torque sensitivity values with respect to velocities and accelerations, but some values would

Table 2 Values of physical parameters. m_j is the link mass [kg], L_j is the link length [cm]

Human model parameter values						
j	1, 6	2, 7	3, 8	4, 9	5, 10	11
m_j	4.5	9.5	3.7	1.2	0.2	12.3
L_j	9	44	43	16	6	49
j	12, 16	13, 17	14, 18	15, 19	20	
m_j	5.8	1.9	1.3	0.5	8	
L_j	17	28	26	19	35	

Note that the index j in this table corresponds to the indexes in columns four-five in Table 1

be zero. For example, for torque and joint angles from different branches, such as $i = 42$ and $m = 18$ in Fig. 3, the torque sensitivity is zero because these branches do not influence each other. The root mean squared (RMS) error was also calculated. The simulations were performed on a ThinkPad notebook with an Intel Core i5-10310U CPU processor and 16 GB RAM. The sensitivity analysis results are provided in the next section.

6 Results

The results for randomly selected torque sensitivity values $\partial Q_{20}/\partial q_5$, $\partial Q_{30}/\partial \dot{q}_{25}$, and $\partial Q_{42}/\partial \ddot{q}_{22}$ are shown in Fig. 4 with the obtained sensitivity trajectories in the left column and the difference between the numerical and analytical methods in the right column. In other words, the error plots on the right show a simple difference between analytical and numerical sensitivities. The results obtained by using the analytical method agree well with the numerical results. The example where the numerical and analytical results differ is shown in Fig. 5. Observe that there are spikes in the numerical torque sensitivity values for $\partial Q_{18}/\partial q_{14}$ at $t = 0.5$ and $t = 1$ s, but similar spikes are not noticeable for $\partial Q_{18}/\partial \dot{q}_{14}$ and $\partial Q_{18}/\partial \ddot{q}_{14}$. This indicates that the observed spike is due to the inaccuracy in the numerical estimation of the sensitivity $\partial Q_{18}/\partial q_{14}$. Otherwise, similar spikes should have been also observed in the plots of velocity and acceleration sensitivities. To further analyze the numerical behavior, we plotted the trajectories of the generalized coordinates q_5 and q_{14}, as well as computed the torques Q_{20} and Q_{18} in Fig. 6. One can see that the numerical errors observed in the sensitivity $\partial Q_{18}/\partial q_{14}$ occur exactly when $q_{14} = 0$ and $Q_{18} = 0$. Thus, numerical errors occur exactly when both the numerator and the denominator vanish. From the trajectories of q_5 and Q_{20}, we observe that, when the coordinate $q_5 = 0$, the torque $Q_{20} \neq 0$. Similarly, the trajectories of $\dot{q}_{25}$ and Q_{30} demonstrate that at times when the former (angular velocity) is zero, the latter (torque) differs from zero, (see Fig. 6). As expected, the corresponding numerical sensitivity $\partial Q_{30}/\partial \dot{q}_{25}$, Fig. 4, is computed correctly. Thus, the numerical algorithm copes well with the sensitivity calculation when only the denominator vanishes or is close to zero, but it might give erroneous results when both the numerator and the denominator are close to zero. Therefore, numerical evaluation of the sensitivity values close to a singularity should be exercised with care. The cumulative time for obtaining all of the analytical sensitivity expressions (measured using `tic` and `toc` commands at each sampling time and summed up for all 301 sampling times) was 16.6 ms, whereas the cumulative time for computing the numerical sensitivity was $2.62 \cdot 10^5$ s, amounting roughly to 3 days and 48 min. Thus, the numerical sensitivity analysis takes seven orders of magnitude more time than its analytical counterpart. We expect that for more complex, nonlinear systems with a larger number of DOF the magnitude of errors and the computational time for the numerical sensitivity analysis would be even larger. These results suggest that the numerical sensitivity analysis is error-prone, time-consuming, and computationally expensive. As a result, the numerical sensitivity analysis should be used only as the very last option when other methods are not available for some reason. Our proposed recursive sensitivity analysis is accurate and efficient.

7 Conclusions

The first-order recursive sensitivity analysis based on the application of the direct differentiation method to the inverse Lagrangian dynamics of rigid multibody systems was presented. We suggested a new description of the system topology based on the branch connectivity graph. The inverse dynamic algorithm based on the Lagrangian formulation was extended to systems with external forces and torques acting at each joint. The advantages of the proposed sensitivity analysis are the following. The method does not suffer from numerical issues associated with the perturbation magnitude in design variables. It has low computational costs and is easy to implement in high-level programming languages such as MATLAB which makes programming and debugging easy. The method is valid for open-chain and open-tree type topological systems. Finally, it is stable, accurate, and numerically efficient. As a result, the proposed algorithm becomes a useful tool in the sensitivity analysis of mechanical multibody systems.

Future work could include the incorporation of the divide-and-conquer algorithm into the sensitivity analysis to speed up computation time through parallelization. An extension of our sensitivity analysis for closed-

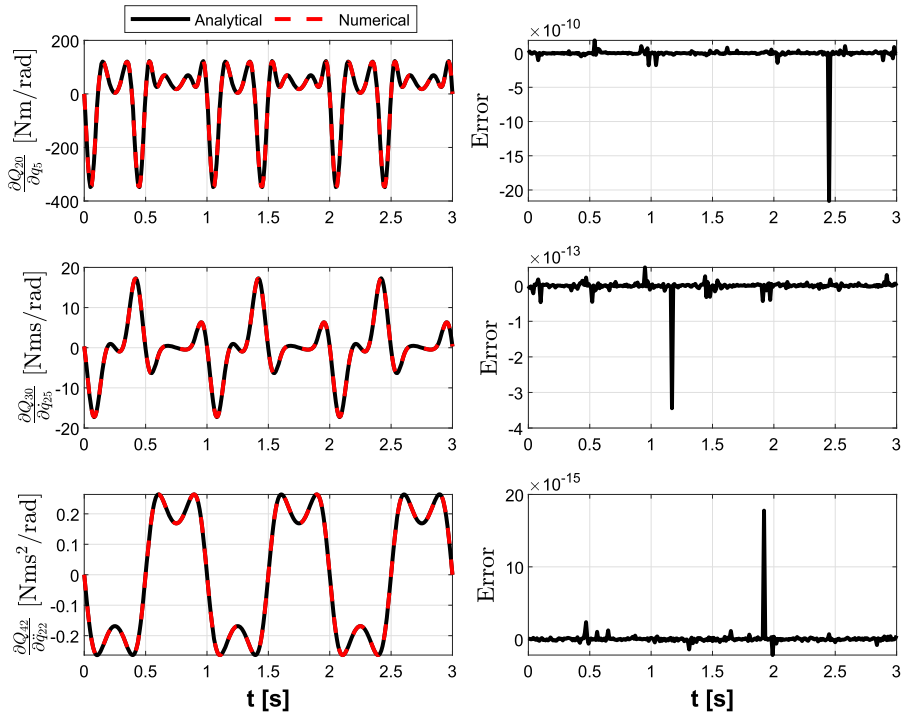

Fig. 4 Results of the sensitivity analysis for $\frac{\partial Q_{20}}{\partial q_5}$, $\frac{\partial Q_{30}}{\partial \dot{q}_{25}}$, and $\frac{\partial Q_{42}}{\partial \ddot{q}_{22}}$

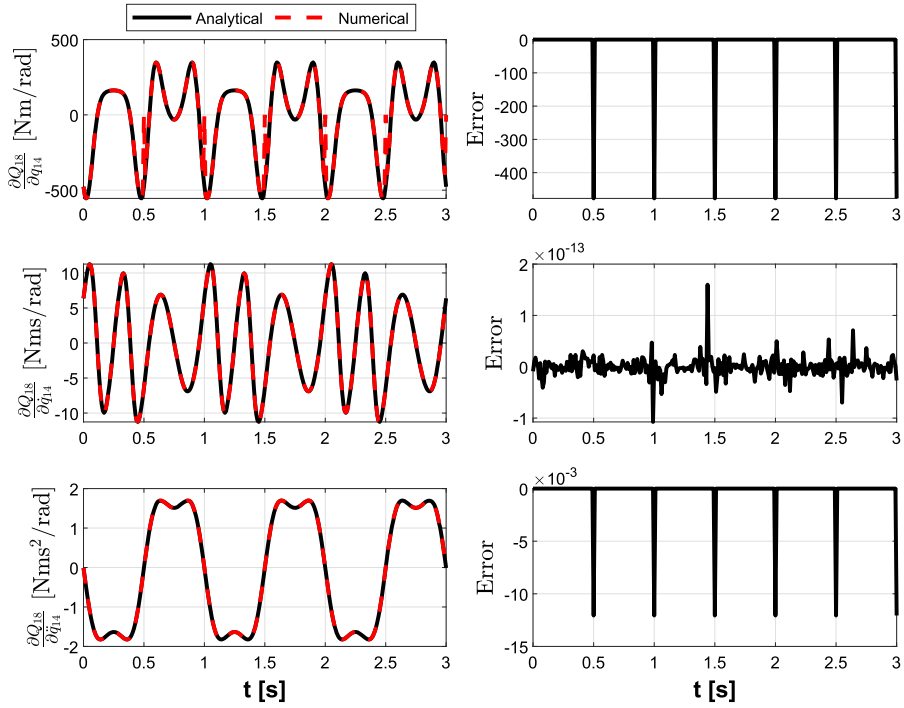

Fig. 5 Results of the sensitivity analysis for $\frac{\partial Q_{18}}{\partial q_{14}}$, $\frac{\partial Q_{18}}{\partial \dot{q}_{14}}$, and $\frac{\partial Q_{18}}{\partial \ddot{q}_{14}}$

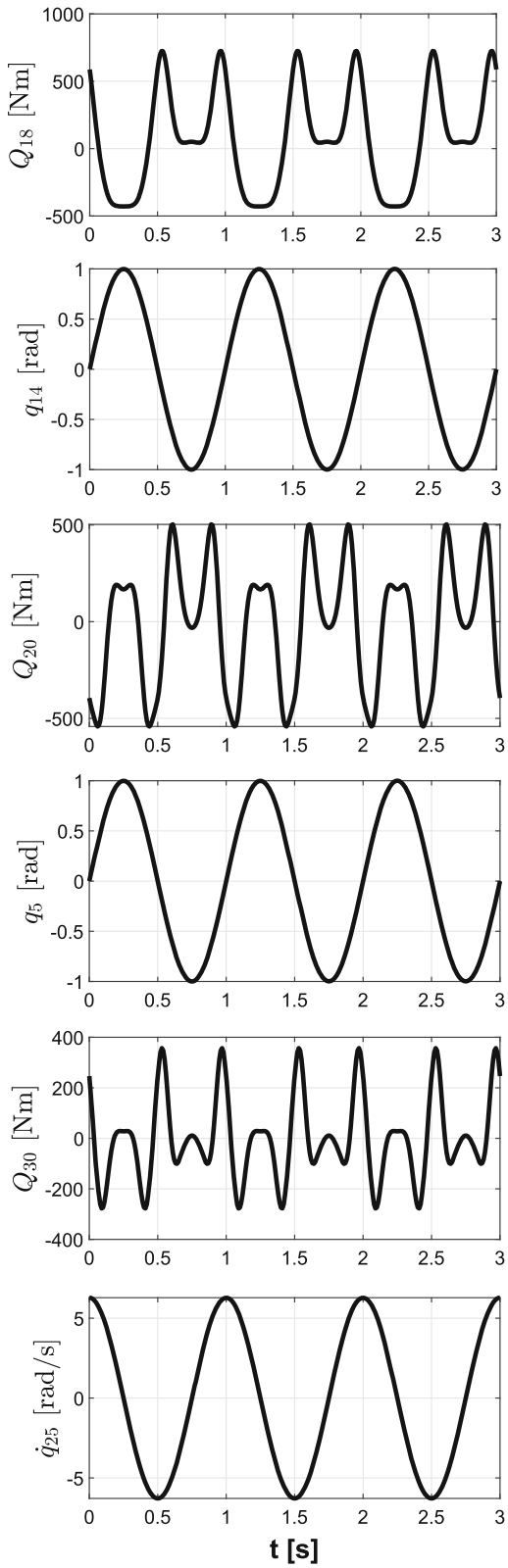

Fig. 6 Time variation of the generalized coordinates q_5, q_{14}, velocity $\dot{q}_{25}$, and torques Q_{18}, Q_{20}, Q_{30}

loop mechanical systems and its application to the human body motion synthesis problem are pending.

Author contributions All authors contributed to writing the manuscript. Simulations were performed by AZ. All authors read and approved the final manuscript.

Funding Open access funding provided by University of Agder. This work has been carried out within the scope of CareWell project funded by the Research Council of Norway under the Grant number 300638.

Data and code availability The MATLAB codes used to generate the results presented in the manuscript are available upon request from the corresponding author.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose. The authors declare that there is no conflict of interests regarding the publication of this article.

References

1. Abdel-Malek, K., Arora, J.S.: Human Motion Simulation: Predictive Dynamics. Elsevier, New York (2013)
2. Anderson, F.C., Pandy, M.G.: A dynamic optimization solution for vertical jumping in three dimensions. Comput. Methods Biomech. Biomed. Eng. **2**(3), 201–231 (1999)
3. Anderson, K.S., Hsu, Y.: Analytical fully-recursive sensitivity analysis for multibody dynamic chain systems. Multibody Syst. Dyn. **8**(1), 1–27 (2002)
4. Azari Nejat, A., Moghadasi, A., Held, A.: Adjoint sensitivity analysis of flexible multibody systems in differential-algebraic form. Comput. Struct. **228**, 106148 (2020)
5. Banerjee, J.M., McPhee, J.: Symbolic sensitivity analysis of multibody systems, pp. 123–146. Springer Netherlands, Dordrecht (2013)
6. Banerjee, J.M., McPhee, J.J.: Graph-theoretic sensitivity analysis of multibody systems. J. Comput. Nonlinear Dyn. **9**(4), 041009 (2014)

7. Bestle, D., Eberhard, P.: Analyzing and optimizing multibody systems. Mech. Struct. Mach. **20**(1), 67–92 (1992)
8. Bhalerao, K.D., Poursina, M., Anderson, K.S.: An efficient direct differentiation approach for sensitivity analysis of flexible multibody systems. Multibody Syst. Dyn. **23**(2), 121–140 (2010)
9. Callejo, A., de Jalón, J.G.: A hybrid direct-automatic differentiation method for the computation of independent sensitivities in multibody systems. Int. J. Numer. Methods Eng. **100**(12), 933–952 (2014)
10. Callejo, A., Dopico, D.: Direct sensitivity analysis of multibody systems: a vehicle dynamics benchmark. J. Comput. Nonlinear Dyn. **14**(2), 021004 (2019)
11. Chadaj, K., Malczyk, P., Frączek, J.: A parallel recursive Hamiltonian algorithm for forward dynamics of serial kinematic chains. IEEE Trans. Robot. **33**(3), 647–660 (2017)
12. Craig, J.J.: Introduction to Robotics. Mechanics and Control. Pearson Prentice Hall, Hoboken (2005)
13. De Jalón, J.G., Unda, J., Avello, A.: Natural coordinates for the computer analysis of multibody systems. Comput. Methods Appl. Mech. Eng. **56**(3), 309–327 (1986)
14. D'Errico, J.: Adaptive robust numerical differentiation (2022). https://se.mathworks.com/matlabcentral/fileexchange/13490-adaptive-robust-numerical-differentiation
15. Dias, J., Pereira, M.: Sensitivity analysis of rigid-flexible multibody systems. Multibody Syst. Dyn. **1**(3), 303–322 (1997)
16. Ding, J.Y., Pan, Z.K., Chen, L.Q.: Second-order sensitivity analysis of multibody systems described by differentialz/algebraic equations: adjoint variable approach. Int. J. Comput. Math. **85**(6), 899–913 (2008)
17. Dopico, D., González, F., Luaces, A., Saura, M., García-Vallejo, D.: Direct sensitivity analysis of multibody systems with holonomic and nonholonomic constraints via an index-3 augmented Lagrangian formulation with projections. Nonlinear Dyn. **93**(4), 2039–2056 (2018)
18. Dopico, D., Zhu, Y., Sandu, A., Sandu, C.: Direct and adjoint sensitivity analysis of ordinary differential equation multibody formulations. J. Comput. Nonlinear Dyn. **10**(1), 011012 (2015)
19. Ezati, M., Ghannadi, B., McPhee, J.: A review of simulation methods for human movement dynamics with emphasis on gait. Multibody Syst. Dyn. **47**(3), 265–292 (2019)
20. Fang, A.C., Pollard, N.S.: Efficient synthesis of physically valid human motion. ACM Trans. Graph. **22**(3), 417–426 (2003)
21. Featherstone, R.: Robot Dynamics Algorithms, 1st edn. Springer, Berlin (1987)
22. Featherstone, R.: A beginner's guide to 6-D vectors (Part 1). IEEE Robot. Auto. Mag. **17**(3), 83–94 (2010)
23. Featherstone, R.: A beginner's guide to 6-D vectors (Part 2) [tutorial]. IEEE Robot. Auto. Mag. **17**(4), 88–99 (2010)
24. Fu, K.S., Gonzalez, R.C., Lee, C.S.G.: Robotics: Control, Sensing, Vision and Intelligence, 5th edn. McGraw-Hill Book Company (1987)
25. Garofalo, G., Ott, C., Albu-Schäffer, A.: On the closed form computation of the dynamic matrices and their differentiations. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2364–2359 (2013)
26. Goldstein, H.: Classical Mechanics. Addison-Wesley (1980)
27. Haug, E.J.: Design sensitivity analysis of dynamic systems. In: Computer Aided Optimal Design: Structural and Mechanical Systems, pp. 705–755. Springer, Berlin (1987)
28. Haug, E.J., Arora, J.S.: Design sensitivity analysis of elastic mechanical systems. Comput. Methods Appl. Mech. Eng. **15**(1), 35–62 (1978)
29. Held, A., Knüfer, S., Seifried, R.: Structural sensitivity analysis of flexible multibody systems modeled with the floating frame of reference approach using the adjoint variable method. Multibody Syst. Dyn. **40**(3), 287–302 (2017)
30. Hollerbach, J.M.: A recursive Lagrangian formulation of maniputator dynamics and a comparative study of dynamics formulation complexity. IEEE Trans. Syst. Man Cyber. **10**(11), 730–736 (1980)
31. Hsu, Y., Anderson, K.S.: Low operational order analytic sensitivity analysis for tree-type multibody dynamic systems. J. Guidance Control Dyn. **24**(6), 1133–1143 (2001)
32. Hwang, R.S., Haug, E.J.: Topological analysis of multibody systems for recursive dynamics formulations. Mech. Struct. Mach. **17**(2), 239–258 (1989)
33. Jain, A.: Robot and Multibody Dynamics. Springer, New York (2010)
34. Jain, A.: Graph theoretic foundations of multibody dynamics. Multibody Syst. Dyn. **26**(3), 307–333 (2011)
35. Jain, A.: Multibody graph transformations and analysis. Nonlinear Dyn. **67**(4), 2779–2797 (2012)
36. de Jalón, J.G., Bayo, E.: Kinematic and Dynamic Simulation of Multibody Systems. Springer, New York (1994)
37. Kim, H., Cho, M.: Study on the design sensitivity analysis based on complex variable in eigenvalue problem. Finite Elements Anal. Des. **45**(12), 892–900 (2009)
38. Laulusa, A., Bauchau, O.A.: Review of Classical Approaches for Constraint Enforcement in Multibody Systems. J. Comput. Nonlinear Dyn. **3**(1) (2007)
39. Lee, L.F., Umberger, B.R.: Generating optimal control simulations of musculoskeletal movement using OpenSim and MATLAB. Peer J **4**, 1–18 (2016)
40. Liu, L., Ballard, D.: Humans use minimum cost movements in a whole-body task. Sci. Rep. **11**(1), 2045–2322 (2021)
41. Luh, J.Y.S., Walker, M.W., Paul, R.P.C.: On-line computational scheme for mechanical manipulators. J. Dyn. Syst. Measure. Control **102**(2), 69–76 (1980)
42. McPhee, J.J.: On the use of linear graph theory in multibody system dynamics. Nonlinear Dyn. **9**(1), 73–90 (1996)
43. Mukherjee, R.M., Bhalerao, K.D., Anderson, K.S.: A divide-and-conquer direct differentiation approach for multibody system sensitivity analysis. Struct. Multidiscip. Opt. **35**(5), 413–429 (2008)
44. 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), 893–904 (2015)
45. Neunert, M., Giftthaler, M., Frigerio, M., Semini, C., Buchli, J.: Fast derivatives of rigid body dynamics for control, optimization and estimation. In: 2016 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR), pp. 91–97 (2016)
46. Peng, H., Zhang, M., Zhang, L.: Semi-analytical sensitivity analysis for multibody system dynamics described by differential-algebraic equations. AIAA J. **59**(3), 893–904 (2021)

47. Pi, T., Zhang, Y., Chen, L.: First order sensitivity analysis of flexible multibody systems using absolute nodal coordinate formulation. Multibody Syst. Dyn. **27**(2), 153–171 (2012)
48. Rodriguez, G., Jain, A., Kreutz-Delgado, K.: A spatial operator algebra for manipulator modeling and control. Int. J. Robot. Res. **10**(4), 371–381 (1991)
49. Serban, R., Haug, E.J.: Kinematic and kinetic derivatives in multibody system analysis. Mech. Struct. Mach. **26**(2), 145–173 (1998)
50. Sohl, G.A., Bobrow, J.E.: A recursive multibody dynamics and sensitivity algorithm for branched kinematic chains. J. Dyn. Syst. Measure. Control **123**(3), 391–399 (2000)
51. Stephen T. Thornton, J.B.M.: Classical dynamics of particles and systems, 5th edn. Thomson Brooks/Cole (2004)
52. Tedrake, R., The Drake Development Team: Drake: Model-based design and verification for robotics (2019). https://drake.mit.edu
53. Tortorelli, D.A., Michaleris, P.: Design sensitivity analysis: overview and review. Inverse Prob. Eng. **1**(1), 71–105 (1994)
54. Tu, T., Wang, G., Rui, X., Zhou, Q., Miao, Y.: Direct differentiation method for sensitivity analysis based on transfer matrix method for multibody systems. Int. J. Numer. Methods Eng. **115**(13), 1601–1622 (2018)
55. van Keulen, F., Haftka, R., Kim, N.: Review of options for structural design sensitivity analysis. Part 1: linear systems. Comput. Methods Appl. Mech. Eng. **194**(30), 3213–3243 (2005)
56. Wojtusch, J., Kunz, J., von Stryk, O.: MBSlib-an efficient multibody systems library for kinematics and dynamics simulation, optimization and sensitivity analysis. IEEE Robot. Auto. Lett. **1**(2), 954–960 (2016)
57. Xiang, Y., Arora, J.S., Abdel-Malek, K.: Optimization-based motion prediction of mechanical systems: sensitivity analysis. Struct. Multidiscip. Opt. **37**(6), 595–608 (2008)
58. Xiang, Y., Arora, J.S., Rahmatalla, S., Abdel-Malek, K.: Optimization-based dynamic human walking prediction: one step formulation. Int. J. Numer. Methods Eng. **79**(6), 667–695 (2009)