2005

On the adjoint formulation of design sensitivity analysis of multibody dynamics cs

Andrei Serban Schaffer
University of Iowa

ON THE ADJOINT FORMULATION OF DESIGN SENSITIVITY ANALYSIS OF MULTIBODY DYNAMICS

by

Andrei Serban Schaffer

An Abstract

Of a thesis submitted in partial fulfillment of the
requirements for the Doctor of Philosophy
degree in Mechanical Engineering
in the Graduate College of
The University of Iowa

December 2005

Thesis Supervisors: Professor James F. Cremer
Professor Lea-Der Chen

ABSTRACT

Numerical methods for design sensitivity analysis of multibody dynamics are presented. An analysis of the index-3 adjoint differential-algebraic equations is conducted and stability of the integration of the adjoint differential-algebraic equations in the backward direction is proven.

Stabilized index-1 formulations are presented and convergence of backward differentiation formulas is shown for the stabilized index-1 forms of the differential-algebraic equations of motion, the direct differentiation differential-algebraic equations, and the adjoint differential-algebraic equations for Cartesian non-centroidal multibody systems with Euler parameters. Convergence of backward differentiation formulas applied to these formulations is proven, by showing that the resulting differential-algebraic equations are uniform index-1.

A novel numerical algorithm is presented, the Piecewise Adjoint method, which formulates the coordinate partitioning underlying ordinary differential equations, resulting from the adjoint sensitivity analysis, as a multiple shooting boundary value problem. The columns of the fundamental matrix and the particular solution of the coordinate partitioning underlying ordinary differential equations are evaluated independently.

Numerical experiments with the Direct Differentiation method, the Adjoint method, and the Piecewise Adjoint method and efficiency analysis are presented for two multibody system models: a four bodies spatial slider-crank and a thirteen bod-

ies High Mobility Multipurpose Wheeled Vehicle. Sequential and parallel numerical experiments validate the correctness of the implementation. The predictions of the number of floating-point operations are confirmed by the sequential results. The predicted speed-up of the parallel numerical experiments is shown for multibody systems with small degrees of freedom and potential speed-ups are discussed for larger problems on architectures with adequate numbers of processors.

Abstract Approved: _____
Thesis Supervisor

_____
Title and Department

_____
Date

_____
Thesis Supervisor

_____
Title and Department

_____
Date

ON THE ADJOINT FORMULATION OF DESIGN SENSITIVITY ANALYSIS OF MULTIBODY DYNAMICS

by

Andrei Serban Schaffer

A thesis submitted in partial fulfillment of the
requirements for the Doctor of Philosophy
degree in Mechanical Engineering
in the Graduate College of
The University of Iowa

December 2005

Thesis Supervisors: Professor James F. Cremer
Professor Lea-Der Chen

Graduate College
The University of Iowa
Iowa City, Iowa

CERTIFICATE OF APPROVAL

PH.D. THESIS

This is to certify that the Ph.D. thesis of

Andrei Serban Schaffer

has been approved by the Examining Committee for the thesis requirement for the Doctor of Philosophy degree in Mechanical Engineering at the December 2005 graduation.

Thesis Committee: _____
James F. Cremer, Thesis Supervisor

_____
Lea-Der Chen, Thesis Supervisor

_____
Kendall E. Atkinson

_____
Suely P. Oliveira

_____
David E. Stewart

ACKNOWLEDGEMENTS

I would like to express my appreciation and gratitude to my advisers, Professors James Cremer and L.D. Chen for their guidance throughout the course of this work. Many thanks are also extended to Professor E.J. Haug, Dr. Radu Serban, Dr. Dan Negrut, Dr. Gisli Ottarson, and Dr. Dale Holtz for their helpful comments and challenging discussions.

Heartfelt thanks must go to my wife and to my parents and for their love and continual support.

ABSTRACT

Numerical methods for design sensitivity analysis of multibody dynamics are presented. An analysis of the index-3 adjoint differential-algebraic equations is conducted and stability of the integration of the adjoint differential-algebraic equations in the backward direction is proven.

Stabilized index-1 formulations are presented and convergence of backward differentiation formulas is shown for the stabilized index-1 forms of the differential-algebraic equations of motion, the direct differentiation differential-algebraic equations, and the adjoint differential-algebraic equations for Cartesian non-centroidal multibody systems with Euler parameters. Convergence of backward differentiation formulas applied to these formulations is proven, by showing that the resulting differential-algebraic equations are uniform index-1.

A novel numerical algorithm is presented, the Piecewise Adjoint method, which formulates the coordinate partitioning underlying ordinary differential equations, resulting from the adjoint sensitivity analysis, as a multiple shooting boundary value problem. The columns of the fundamental matrix and the particular solution of the coordinate partitioning underlying ordinary differential equations are evaluated independently.

Numerical experiments with the Direct Differentiation method, the Adjoint method, and the Piecewise Adjoint method and efficiency analysis are presented for two multibody system models: a four bodies spatial slider-crank and a thirteen bod-

ies High Mobility Multipurpose Wheeled Vehicle. Sequential and parallel numerical experiments validate the correctness of the implementation. The predictions of the number of floating-point operations are confirmed by the sequential results. The predicted speed-up of the parallel numerical experiments is shown for multibody systems with small degrees of freedom and potential speed-ups are discussed for larger problems on architectures with adequate numbers of processors.

TABLE OF CONTENTS

LIST OF TABLES

Table

LIST OF FIGURES

Figure

LIST OF ABBREVIATIONS

ADAE	The Adjoint Differential-Algebraic Equations
BDF	Backward Differentiation Formulas
BVP	Boundary Value Problem
CPUODE	Coordinate Partitioning Underlying Ordinary Differential Equations
DAE	Differential-Algebraic Equations
EUODE	Essential Underlying Ordinary Differential Equations
HMMWV	High Mobility Multipurpose Wheeled Vehicle
IVP	Initial Value Problem
ODE	Ordinary Differential Equations

LIST OF SYMBOLS

β	The vector of design parameters
Φ	The constraint function vector
λ	The function vector of Lagrange multipliers of the DAE of motion
μ	The Adjoint variable function vector
ν	The function vector of Lagrange multipliers of the Adjoint DAE
Ψ	The design sensitivity analysis functional
τ_X^Y	The execution time of algorithm Y of task X
$C^k(\Omega)$	The space of functions with continous derivatives up to k-th order on Ω
M	The mass matrix
L	The Coriolis term
$\mathcal{N}(C)$	The null-space of matrix C
q	The generalized coordinates function vector
Q	The function vector of applied forces

CHAPTER 1
INTRODUCTION

The continuous development of computation power and cost decrease of computer hardware, especially of internal memory and microprocessor, account for increasing accuracy of mathematical models being used to represent physical phenomena, such as the dynamics of a mechanical system. The computation power currently available benefits from microprocessor performances and number of memory accesses per time unit that have increased almost exponentially since the mid 1980s [29]. However, the variety of computer architectures currently available, especially multiprocessors architectures, present special challenges for the modeler; e.g., interprocess communication and synchronization, process and thread management and model formulation differing from the uniprocessor model formulation.

The modeling of mechanical systems often requires the study of dependency of a model on parameters whose values cannot be accurately known. Among the purposes of modeling a multibody system are those of model optimization, parameter estimation, model simplification, data assimilation, optimal control, process sensitivity, and making a design less sensitive to variation occurring due to the manufacturing process. All of these goals imply computation, over a finite time-interval, of the evolution of the multibody system generalized coordinate vector and its derivatives with respect to model parameters. The goal of this thesis is to develop new ways of using modern multiprocessor architectures to solve the various aspects of the design sensitivity analysis of multibody systems. Although efficient ways of solving for sen-

sitivities of multibody system response have been defined [26] and well-established computer software is in a stable and mature phase [37], there are still design sensitivity analysis aspects that remain unsolved or algorithms that do not exploit the benefits of multiprocessor architectures.

The Direct Differentiation formulation of design sensitivity analysis of multibody systems [26] is well-suited for parallel computation, but it requires the integration of a number of differential-algebraic equations (DAE), the Direct Differentiation DAE, equal to the number of model parameters plus one (for integrating the DAE of motion). Although each such system can be solved on a different processor, the number of necessary processors is as large as the number of model parameters, which can be substantial.

In the Adjoint formulation [26] derivatives of generalized coordinates with respect to model parameters are not computed directly. Rather, all the terms required for assembling the gradients of functionals that typically occur in design sensitivity analysis, optimization, or optimal control are computed by solving only one DAE, the Adjoint DAE [26], in addition to the DAE of motion.

One of the disadvantages of the Adjoint method is the necessity of backward integration of the Adjoint DAE, which must be started at the end of the integration of the DAE of motion. As a result, generalized coordinates of the equations of motion must be stored during integration of the DAE of motion, in order to be used for constructing and integrating the Adjoint DAE backward in time, after the final time is reached. By doing so, a heavy load is imposed on memory resources of the computing

system. Furthermore, there is a potential loss in accuracy, due to the fact that the generalized coordinates cannot be stored at each step of the integration of the DAE of motion, and therefore have to be interpolated between the storage steps. Also, the Adjoint method does not take full advantage of a parallel architecture, since the computation flow has an inherently sequential structure, due to the fact that the initial time for the Adjoint DAE coincides with final time for the DAE of motion.

A modified Adjoint method is presented, in which computation can be distributed to $2d + 2$ processors, where d is the number of degrees of freedom of the multibody system, and the Adjoint backward initial value problem (IVP) is replaced by a boundary value problem (BVP), which is solved through a backward multiple shooting algorithm [4]. In the equivalent BVP formulation, each time-step is solved in the backward direction, but time-steps advance in the forward direction. As a result, the memory load for storing generalized coordinates during integration of the DAE of motion, is minimized.

1.1 Parallel Granularity

The efficiency of parallel methods depend on the amount of data communication and synchronization between parallel modules, compared with the amount of computation in each module. The dependency between the amount of work and the amount of synchronization in parallel modules is measured by the *granularity* of the parallelism [49].

Coarse and fine granularity are relative measures that depend on the size of

the problem to be solved, the multiprocessor architecture, and the operating system's process management and interprocess communication primitives [54], [15], [45]. In this thesis, coarse-grained parallel routines are considered those routines that solve for at least one step of a differential system of equations, routines that compute solutions of linear systems of equations, or routines that iteratively solve non-linear systems of equations as part of the numerical integration of a system of differential equations.

Fine-grained parallel methods have a substantial amount of data communication and synchronization between parallel modules, compared to the average amount of actual computation that is done by the modules. In order to be efficient, fine-grained parallelism may require a computing model; e.g., *cellular automata*, different from the Turing computing model [10] or they may require solving the problem using languages [12] especially designed to make fine-grained parallelism efficient. Efficient fine-grained parallelism is an ongoing Computer Science research problem and is not a research goal of this thesis.

1.2 Thesis Objectives

The overall objective of this thesis is to reformulate the Adjoint method [26] of design sensitivity analysis such that the numerical algorithm of the new formulation preserves the major advantage of the Adjoint method; i.e., the number of DAE, that the numerical algorithm must integrate, does not depend on the number of design parameters; and has the following advantages of the Direct Differentiation method [26]:

1. The numerical algorithm advances forward in time.

2. The numerical algorithm can be executed by independent threads of computation.

In addition, the new formulation must minimize the memory load required by backward integration of the Adjoint method.

The specific objectives of the thesis are as follows:

1. To prove backward stability of the Adjoint method for the index-3 DAE of motion of a non-centroidal spatial multibody system in which orientation is defined by Euler parameters. In order to develop numerical algorithms for the Adjoint method applied to index-3 DAE, stability of the analytic solution of the Adjoint index-3 DAE [26] in the backward direction must be proven. Backward stability of the analytic solution of the Adjoint DAE has been shown [16] for Adjoint DAE of index-0, index-1, and index-2 in Hessenberg form. The Adjoint DAE of the index-3 DAE of motion of a non-centroidal multibody system with Euler parameters is an index-3 DAE [26] that cannot be brought to Hessenberg form [13], because the highest derivative coefficient, the mass matrix, is singular [52].

2. To prove convergence of the Hiller-Anantharaman [2] stabilized index-1 formulation. Stabilized index-1 DAE formulations are obtained for the index-3 DAE of motion [27], the index-3 Direct Differentiation DAE, and the index-3 Adjoint DAE [26], which are integrated using the Lawrence Livermore National

Laboratory's Implicit Differential-Algebraic (IDA) solver [31].

3. To develop a new method, called the Piecewise Adjoint method, that (1) requires the integration of a number of DAE that does not depend on the number of design parameters; (2) has a parallel computational structure; (3) progresses forward in time; and (4) minimizes the memory load required by the backward integration of the Adjoint DAE.

1.3 Thesis Structure

The thesis is organized as follows: Chapter 2 is a review of literature on computational aspects of design sensitivity analysis of multibody systems. Chapter 3 presents a stability analysis of backward integration of the Adjoint DAE. Chapter 4 presents the Hiller-Anantharaman index-1 formulations of the the index-3 DAE of motion, the index-3 Direct Differentiation DAE, and the index-3 Adjoint DAE and shows convergence results for backward differentiation formulas (BDF) applied to the stabilized index-1 formulations. Chapter 5 presents the Piecewise Adjoint method and efficiency and memory load analysis. Chapter 6 presents results of sequential and parallel numerical experiments and methods of error control. The Appendix presents algorithms for efficiently evaluating the time-derivatives of a matrix; orientation matrices; kinematic matrices required for solving the Adjoint coordinate partitioning underlying ODE of Chapter 5; and partial derivatives of kinematic matrices and force elements. The bibliography ends the thesis.

CHAPTER 2
LITERATURE REVIEW

Numerical methods for multibody system modeling are based on the formulation of differential-algebraic equations (DAE). Because differential-algebraic equations cannot be treated as ordinary differential equations [46], they require a more complex numerical approach.

Conditions for existence and uniqueness of the solution of DAE have been established [13], starting from the analysis of linear time invariant DAE using the matrix-pencil theory, and extending to nonlinear higher index DAE in Hessenberg form (the equations of motion for a multibody system are an example of index-3 DAE in Hessenberg form). Several ways have been investigated to define numerical algorithms for solving DAE. Projection methods work by integrating the differential equations of the corresponding index two or one DAE, and then projecting the solution onto the manifold defined by the constraint equations [35]. State-space methods reduce the DAE to an underlying ODE. By integrating the underlying ODE, only a subset of the generalized coordinates are integrated, the remaining generalized coordinates being recovered from the constraint equations. Examples of such methods are the generalized coordinate partitioning method [58]. Based on the implicit function theorem, a partitioning of the generalized coordinates into independent and dependent coordinates is defined and a corresponding underlying ODE is obtained. Another example of a state-space method is the differential-geometric approach [47], in which the DAE are formulated as differential equations on manifolds. Other approaches to

solving DAE consider an overdetermined system, consisting of the original DAE and one or more of the derivatives of the constraint equations. Examples of such methods are presented by Fuhrer and Leimkuhler [22], where the solution is obtained by the use of a special pseudo-inverse, and Jay [33], where a special Runge-Kutta method is defined to integrate the differential equations of motion of a multibody system, satisfying position, velocity, and acceleration constraints. A different formulation, due to Rabier and Rheinboldt [48] defines the DAE for a multibody system based on the Gauss principle of least constraint. Stiff mechanical systems bring additional complexity to the numerical integration of the DAE, requiring implicit integration methods [42].

Collocation methods can also be used to solve both initial value problems DAE and boundary value problems DAE. Collocation methods based on a regularized boundary value problem approximate the solution by a continuous piecewise polynomial and represent consistent approximations at mesh points by applying Radau schemes to linear variable coefficient DAE of any index [53]. Under weak assumptions, the collocation problems are uniquely and stably solvable and, if the solution is sufficiently smooth, super-convergence at mesh points is shown. Asher and Spiteri [7] describe methods and implementation of a general-purpose code, COLDAE, that can solve boundary value problems for nonlinear systems of semi-explicit DAE of index at most 2 and fully implicit index-1 DAE. The method implemented is piecewise polynomial collocation at Gaussian points, extended as needed by the projection method of Ascher-Petzold [3].

Current implementations of DAE methods such as projection, state-space, differential-geometric, coordinate-partitioning, Runge-Kutta and collocation generally have a formulation in which the computation flow is sequential. Therefore, currently available software packages have a high degree of data dependency that makes them unsuitable for parallel computing architectures.

2.1 Numerical Methods for Design Sensitivity Analysis

Numerical methods and software for sensitivity analysis of DAE [37] have been defined by investigating three approaches to solving the system obtained by combining the original DAE and the sensitivity analysis DAE. The resulting Jacobian can be approximated by a block-diagonal matrix, while retaining rapid Newton convergence and a block-diagonal pre-conditioner is highly effective. Three new codes have been introduced; DASSLSO, DASPKSO, and SENSD. The first two are modifications to the ODE/DAE solvers DASSL and DASPK, respectively. The third code is an auxiliary routine that allows a user to perform sensitivity analysis of a derived quantity; e.g., the L2 norm of the solution vector. The resulting simultaneous corrector method combines the DAE and sensitivities to form a system that is solved using a BDF method at each step. The nonlinear system obtained is then solved by applying Newton's method to the corrector equation, where the Jacobian is approximated by its block diagonal part. This method achieves 2-step quadratic convergence for nonlinear problems, allowing the factored corrector matrix to be reused for mul-

tiple steps. As a result, this method is a significant improvement over the staggered direct method [17], because the need for additional matrix factorizations to solve the sensitivity system has been eliminated.

Feehery et al. [21] have developed a novel corrector method (called the staggered corrector sensitivity method) for solving DAE and sensitivities, which exhibits a smaller computational cost. They use two corrector iteration loops at each step, one for the DAE and the second for sensitivities. Computational savings result from fewer Jacobian updates, in order to evaluate residuals of the sensitivity equations. The staggered corrector sensitivity method is an improvement on the simultaneous corrector method, because the later requires the system Jacobian to be updated at each corrector iteration. Although this cost is minor compared with matrix factorization, it may become significant for large problems.

Error, convergence, and stability analysis play an important role, not only as overall measures of the properties of the algorithm, but also as tools for step-size control, in order to avoid under-solving or over-solving, especially in implicit methods, which require simplified Newton iterations [32]. Stability analysis is necessary to establish whether a given formulation or algorithm is well posed. It has been studied for Adjoint DAE of index-0, index-1, and index-2 [16], in Hessenberg form. Error, convergence, and stability analysis remain to be done for index three DAE problems, since design sensitivity analysis for multibody systems require an index-3 DAE formulation.

2.2 Parallel Methods for Design Sensitivity Analysis of Multibody Systems

Parallel computation of multibody systems can bring several benefits to the task of numerical integration of the corresponding DAE. If the goal is real-time simulation or reduction of the design cycle, pipelining different stages of computation is a way of decreasing computation time. A re-formulation of multibody system equations of motion might generate a system with high data independency, therefore enabling several parts of the computation to be done simultaneously and decreasing computation time. Another motivation of parallel computation is the desire for a more robust and reliable code; i.e., obtaining reliable error estimates and accurate dense output [20].

In order to take advantage of parallel computing capabilities, the multibody system might require a special formulation [11]. Different parallel architectures and operating systems may also require different formulations of the same problem [49]; e.g., implicit Runge-Kutta methods can be redefined by using a pre-conditioner, in order to obtain a parallel structure of the algorithm [34].

The waveform relaxation technique (WR) is an efficient tool for solving large systems of ODE and DAE in multiprocessor environments. The basic idea is to decouple a system of ODE by integrating one equation (or a subset of equations) for one unknown (or a subset of unknowns), with all the other unknowns taken from previous steps. Under mild assumptions, the iterative application of this algorithm

to all unknowns (or subsets of unknowns) will converge to the solution [59].

Waveform relaxation methods for DAE have been investigated by van der Houwen and van der Veen [55]. Three families of algorithms have been defined, as follows:

1. A method suitable for higher index DAE (up to index-3), which does not take advantage of the structure of the system.

2. A partitioned DAE method that can be applied to semi-explicit DAE with faster convergence.

3. A method that can be applied only to index-1 DAE with even faster convergence and enhanced parallelism.

Other waveform relaxation techniques for DAE are based on Runge-Kutta methods and the application of an iterative method that is independent of the number of stages, when implemented on a multiprocessor environment [56]. However, the number of iterations required to achieve convergence is generally substantial. Therefore special algorithms, such as Krylov-subspace acceleration, are necessary to achieve faster convergence, especially for large systems of equations that are typically stiff [36].

Design sensitivity analysis of a multibody system also implies a DAE formulation. Two approaches have been defined for calculating gradients of functionals required in mechanical design, the Direct Differentiation method and the Adjoint method [26]. The Direct Differentiation method requires integrating the multibody

system equations of motion and as many additional independent DAE as the number of model parameters. As a result, although suitable for a parallel computation approach, it may still result in a difficult computation task when the number of model parameters is much larger than the number of available parallel processors. Therefore, the Adjoint method formulation, which requires solving only one additional DAE, might be the only reasonable choice.

However, the Adjoint method requires backward numerical integration and storage of the generalized coordinate vector and its first order derivative at many, if not all integration steps for the multibody system equation of motion [26]. Although computing power is considered orthogonal to storage capability; i.e., one cannot be increased without decreasing the other, multiprocessing can provide a solution for minimizing the storage load and retaining computational efficiency in existing formulations of the Adjoint method. In order to obtain an algorithm suitable for parallelism, a new mathematical formulation must be defined for the problem being modeled. The goal is to minimize data dependency between parallel threads, therefore minimizing the overhead due to interprocess communication and synchronization. An application of the principle of a more complex mathematical formulation of a problem in order to achieve a parallel algorithm is a parallel version of the modified Gram-Schmidt algorithm [57]. A reformulation of the Adjoint method, that can reduce or eliminate data dependency between modules, by independently solving sets of differential equations, is considered a coarse-grained parallel method.

Design sensitivity analysis requires the calculation of multibody system kine-

matic expressions and their derivatives with respect to generalized coordinates and model parameters. In a multibody system Cartesian formulation with Euler parameters, basic identities have been developed for efficient computation of such derivatives [51].

CHAPTER 3
STABILITY OF THE ADJOINT DIFFERENTIAL-ALGEBRAIC EQUATIONS OF INDEX-3 MULTIBODY EQUATIONS OF MOTION

Stability criteria for quasi-linear differential-algebraic equations (DAE), of index up to three have been established [39], based on information about the real part of the spectrum of the matrix coefficient of the highest order derivative term. However, such information is not directly available for the adjoint DAE of the DAE of dynamics. Stability analysis of the analytic solution of a linearized index-m DAE in Hessenberg form has been studied [5], but the adjoint equation of a multibody DAE of motion is not in Hessenberg form, since the coefficient matrix of the highest order derivative term can be singular. Moreover, analysis performed by Ascher and Petzold [5] is not extended to the adjoint equation corresponding to a given DAE, or to the adjoint equation stability properties with respect to those of the given DAE. The relation between stability properties of a given DAE and its adjoint equation is studied in Ref. [16], for DAE of index not greater than two in Hessenberg form.

The purpose of this chapter is to establish stability results for adjoint equations corresponding to semi-explicit index-3 DAE that have a more general form than Hessenberg, particularly index-3 DAE that arise in the study of multibody dynamics. In such index-3 DAE, the highest derivative term has a coefficient matrix that is singular, if Euler parameters are used for body orientation [52].

3.1 Adjoint Equations of Multibody Equations of Motion

Consider a multibody system of n_b rigid bodies, between which there are m_0 joint constraint equations. The position of body i, $i = 1, 2, \ldots, n_b$, is defined in a global Cartesian coordinate frame by a 3×1 vector r_i, and orientation of the body is defined by a 4×1 Euler parameter vector p_i. Therefore, the configuration of a multibody system is described by a vector of $n = 7 \times n_b$ generalized coordinates, $q_{n\times 1} = \begin{pmatrix} {q_1}^\top & \ldots & {q_{n_b}}^\top \end{pmatrix}^\top$, in which the 7×1 vector q_i is $q_i = \begin{pmatrix} {r_i}^\top & {p_i}^\top \end{pmatrix}^\top$, $i = 1, 2, \ldots, n_b$. The multibody system has a number $m = m_0 + n_b$ of algebraic constraints; i.e., the m_0 joint constraint equations plus the n_b Euler parameter normalization constraints, ${p_i}^\top p_i - 1 = 0$, $i = 1, 2, \ldots, n_b$. The equations of motion of such a constrained multibody system are [27]

$$M(q, \beta, t)q'' - L(q, q', \beta, t) - Q(q, q', \beta, t) + {\Phi_q}^\top \lambda = 0 \tag{3.1}$$

$$\Phi(q, \beta, t) = 0 \tag{3.2}$$

in which $'$ denotes the time derivative, $q(\beta, t)_{n\times 1}$ is the vector of generalized coordinates, depending on time t and n_β time independent model parameters $\beta_{n_\beta \times 1}$; $\lambda(\beta, t)_{m\times 1}$ is the vector of Lagrange multipliers; $M(q, \beta, t)_{n\times n}$ is the system mass matrix; $L(q, q', \beta, t)_{n\times 1}$ contains Coriolis and related terms; $Q(q, q', \beta, t)_{n\times 1}$ is the vector of applied forces; and $\Phi(q, \beta, t)_{m\times 1}$ is the constraint function vector. Equations (3.1) and (3.2) represent an index-3 system of differential-algebraic equations [13]. The multibody system described by Eqs. (3.1) and (3.2) has $d = n - m$ degrees of free-

dom. The constraint Jacobian Φ_q is assumed to be twice continuously differentiable with respect to time, $\Phi_q \in C^2(\mathcal{I})$, $t \in \mathcal{I} = [t^1, t^2]$, and bounded $\|\Phi_q\| \leq K_q$. The mass matrix M is assumed to be symmetric and positive definite on the null space of the constraint Jacobian, and vectors q and r are assumed to be twice continuously differentiable with respect to time and bounded.

As a result of linearization of Eqs. (3.1) and (3.2), the following index-3 DAE is obtained [26]:

$$Mq'' = \sum_{j=1}^{2} A_j z_j + B\lambda + s \tag{3.3}$$

$$0 = Cq + r \tag{3.4}$$

in which $z_j(\beta, t) = \frac{\partial q^{(j-1)}}{\partial t^{(j-1)}}$ represents the $(j-1)$-th derivative of the generalized coordinate vector q with respect to time, $B = -{\Phi_q}^\top$, $C = -\Phi_q$, $A_1 = L_q + Q_q - ({\Phi_q}^\top \lambda)_q - (Mq'')_q$, and $A_2 = L_{q'} + Q_{q'}$. In order to obtain an essential underlying ordinary differential equation, Ascher and Petzold show [5] that the change of variable

$$u = Rq \tag{3.5}$$

may be performed, in which the time and model parameter dependent matrix

$$R(\beta, t)_{d \times n}$$

has the properties

$$\|R\| < K \tag{3.6}$$

$$R \in C^2(\mathcal{I}), t \in \mathcal{I} = [t^1, t^2] \tag{3.7}$$

$$rank\ (R) = d \tag{3.8}$$

$$RB = 0 \tag{3.9}$$

Provided that the following assumptions are valid:

1. $A_j(\beta, t), j = 1, 2;\ B, C \in C^2(\mathcal{I})$

2. Matrix $C(\beta, t)$ has full row-rank for any $t \in \mathcal{I}$. As a result, $C(\beta, t)B(\beta, t) = \Phi_q {\Phi_q}^\top$ is nonsingular for any $t \in \mathcal{I}$

3. $\|{A_j}^{(k)}\| < M, j = 1, 2;\ \|B^{(k)}\| < M, \|C^{(k)}\| < M, k = 0, 1, 2$

the matrix $\begin{pmatrix} R \\ C \end{pmatrix}$ is invertible [5], with inverse

$$\begin{pmatrix} R \\ C \end{pmatrix}^{-1} = \begin{pmatrix} S & F \end{pmatrix} \tag{3.10}$$

in which matrix $S_{n\times d}$ is constructed such that

$$RS = I \tag{3.11}$$

$$CS = 0 \tag{3.12}$$

and $F_{n\times m}$ is defined [5] as

$$F = B(CB)^{-1} \tag{3.13}$$

Due to boundedness of the constraint Jacobian Φ_q, matrix F is also bounded. With the additional assumptions that $\|R\| > k_R > 0$ and non-singular matrix $RR^\top$ is well-conditioned; i.e, its condition number is bounded, $\kappa(RR^\top) < K_{RR^\top} < \infty$, matrix S is bounded. This result follows from properties defined by Eqs. (3.11) and (3.12), according to Theorem A.3 in the Appendix.

Since $\begin{pmatrix} R \\ C \end{pmatrix} q = \begin{pmatrix} u \\ -r \end{pmatrix}$, q is obtained as

$$q = \begin{pmatrix} R \\ C \end{pmatrix}^{-1} \begin{pmatrix} u \\ -r \end{pmatrix} = \begin{pmatrix} S & F \end{pmatrix} \begin{pmatrix} u \\ -r \end{pmatrix} = Su - Fr \tag{3.14}$$

Differentiating Eq. (3.14) twice with respect to time yields

$$q'' = Su'' + \sum_{j=0}^{1} \begin{pmatrix} 2 \\ j \end{pmatrix} S^{(2-j)} u^{(j)} - (Fr)'' \tag{3.15}$$

in which parenthesized superscript notation, $^{(j)}$ denotes index j of differentiation. Pre-multiplying Eq. (3.15) with the product RM; pre-multiplying Eq. (3.3) with matrix R, accounting for the property that $RB = 0$; and subtracting term by term the two equations thus obtained, the essential underlying ODE (EUODE) of the index-3 DAE defined by Eqs. (3.3) and (3.4) is

$$\begin{aligned} RMSu'' &= (RA_2S - 2RMS')u' \\ &+ (RA_1S - RMS'' + RA_2S')u + \hat{r} \end{aligned} \tag{3.16}$$

in which $\hat{r} = Rq + RM(Fr)'' - RA_1Fr - RA_2(Fr)'$.

In order for Eq. (3.16) to be an ODE, its coefficient matrix must be non-singular. Since matrix M is assumed to be positive definite on the null-space $\mathcal{N}(C)$ of matrix C, the following theorem holds:

Theorem 3.1. *Consider matrices $T_{1(d\times n)}$ and $T_{2(n\times d)}$ having the following properties:*

1. *T_1T_2 is a non-singular matrix*

2. *$CT_2 = 0$, where $C = \Phi_q$ is the constraint Jacobian matrix*

3. *$T_1B = 0$, where $B = C^\top = {\Phi_q}^\top$*

With assumptions 1 through 3, if the mass matrix M of Eq. (3.3) is positive definite on $\mathcal{N}(C)$; i.e., $\eta^\top M\eta > 0$, for any $\eta \neq 0$ such that $C\eta = 0$, then matrix T_1MT_2 is non-singular.

Proof. Assume that matrix $(T_1MT_2)_{d\times d}$ is singular. Then, there is a vector $\xi \in \mathbb{R}^d$, $\xi \neq 0$, such that $T_1MT_2\xi = 0$. Letting $\eta = T_2\xi$, then $C\eta = CT_2\xi = 0$. Therefore, $\eta \in \mathcal{N}(C)$. If vector η is zero, then $T_1\eta = T_1T_2\xi = 0$, but since T_1T_2 is non-singular, vector ξ must also be zero, which contradicts the assumption $\xi \neq 0$. Therefore, η is a non-zero vector. Let $\zeta = M\eta$. Vector ζ is also a non-zero vector, because $\eta^\top\zeta = \eta^\top M\eta \neq 0$, since M is positive definite on $\mathcal{N}(C)$, and non-zero vector η belongs to the null-space $\mathcal{N}(C)$ of matrix C. Using the vectors η and ζ defined above, $T_1MT_2\xi = 0$ is re-written as

$$T_1MT_2\xi = T_1M\eta = T_1\zeta = 0$$

Matrix $(C^\top)_{n\times m}$ is

$$C^\top = \begin{pmatrix} c_1 & \dots & c_m \end{pmatrix} \tag{3.17}$$

Since C has full row-rank, its rows $\{c_j\}_{j=1,2,\dots,m}$, which are columns of $C^\top$, are linearly independent. Let $\mathcal{B}_n \equiv \{c_1, \dots, c_m, \bar{c_1}, \dots, \bar{c_d}\}$ be a basis in $\mathbb{R}^n$. Therefore, vector $\zeta \in \mathbb{R}^n$ can be written as a linear combination of vectors in the basis $\mathcal{B}_n$,

$$\zeta = \sum_{j=1}^{m} \alpha_j c_j + \sum_{k=1}^{d} \rho_k \bar{c_k} = C^\top \alpha + \bar{C}\rho \tag{3.18}$$

in which $\bar{C}_{n\times d} \equiv \begin{pmatrix} \bar{c_1} & \dots & \bar{c_d} \end{pmatrix}$, $\rho = \begin{pmatrix} \rho_1 & \dots & \rho_d \end{pmatrix}^\top$, and $\alpha = \begin{pmatrix} \alpha_1 & \dots & \alpha_m \end{pmatrix}^\top$. Assume that there is at least one $\rho_k \neq 0$, for some $k \in \{1, 2, \dots, d\}$; i.e., ρ is a non-zero vector. Pre-multiplying Eq. (3.18) with matrix T_1 and accounting for the fact that $T_1 B = T_1 C^\top = 0$,

$$0 = T_1 \zeta = T_1 \bar{C} \rho \tag{3.19}$$

Consider each column s_j, $j = 1, 2, \dots, n$, of matrix $T_2 = \begin{pmatrix} s_1 & \dots & s_d \end{pmatrix}$, re-written as a linear combination of the vectors in the basis $\mathcal{B}_n$,

$$s_l = \sum_{j=1}^{m} \tau_{l,j} c_j + \sum_{k=1}^{d} \gamma_{l,k} \bar{c_k} = C^\top \tau_l + \bar{C}\gamma_l \tag{3.20}$$

in which $\tau_l = \begin{pmatrix} \tau_{l,1} & \dots & \tau_{l,m} \end{pmatrix}^\top$ and $\gamma_l = \begin{pmatrix} \gamma_{l,1} & \dots & \gamma_{l,d} \end{pmatrix}^\top$, $l \in \{1, 2, \dots, d\}$. By defining matrices $T_{m\times d} \equiv \begin{pmatrix} \tau_1 & \dots & \tau_d \end{pmatrix}$ and $\Gamma_{d\times d} \equiv \begin{pmatrix} \gamma_1 & \dots & \gamma_d \end{pmatrix}$, matrix T_2 may be re-written as

$$T_2 = C^\top T + \bar{C}\Gamma \tag{3.21}$$

Pre-multiplying Eq. (3.21) with T_1 and accounting for the property that

$$T_1 C^\top = T_1 B = 0$$

the following identity is obtained:

$$T_1 T_2 = T_1 \bar{C} \Gamma \tag{3.22}$$

As a result, matrix $T_1\bar{C}\Gamma$ is non-singular. Applying the Sylvester inequality [23]; according to which for two matrices $M_{1(m\times n)}$ and $M_{2(n\times p)}$,

$$rank(M_1 M_2) \leq min(rank(M_1), rank(M_2))$$

to the matrix product $T_1\bar{C} \cdot \Gamma$, the rank of matrix product $T_1\bar{C}\Gamma$ cannot exceed the rank of $T_1\bar{C}$; i.e., $d = rank(T_1\bar{C}\Gamma) \leq rank(T_1\bar{C})$. As a result, matrix $T_1\bar{C}$ is also non-singular. Therefore, in Eq. (3.19), vector ρ must be zero. Consequently, vector ζ in Eq. (3.18) is rewritten as $\zeta = C^\top \alpha$. Pre-multiplying vector ζ with ${T_2}^\top$, the following identity is obtained:

$${T_2}^\top \zeta = {T_2}^\top M \eta = {T_2}^\top C^\top \alpha = (CT_2)^\top \alpha = 0 \tag{3.23}$$

where $CT_2 = 0$ is used. Therefore,

$$\eta^\top M \eta = \xi^\top {T_2}^\top M \eta = 0 \tag{3.24}$$

Since η is a non-zero vector belonging to the null-space of matrix C, Eq. (3.24) contradicts the hypothesis that M is positive definite in $\mathcal{N}(C)$. Therefore, $\xi \neq 0$ cannot be true, and matrix $T_1 M T_2$ is non-singular. ■

Corollary 3.2. *Matrix RMS is non-singular, where M is the mass matrix, R has the properties of Eqs.* (3.6) *through* (3.9)*, and S was constructed such that $RS = I$ and $CS = 0$.*

Proof. Matrices R and S have the properties of T_1 and T_2 respectively, since $RS = I$, $RB = 0$, and $CS = 0$ and the mass matrix is positive definite on the null-space of the constraint Jacobian. Therefore matrix RMS, according to Theorem 3.1, is non-singular. ■

3.2 Stability Analysis of the Adjoint DAE of the DAE of Motion

Consider the formulation of a system of ordinary differential or differential - algebraic equations

$$\mathcal{F}(x', x, t, \beta) = 0 \tag{3.25}$$

The Jacobian $\mathcal{F}_{x'}$ is non-singular if the system of Eq. (3.25) is an ODE, and singular if the system is a DAE. The adjoint system corresponding to Eq. (3.25) that is used for computing sensitivity $\frac{dG}{d\beta}$ of an objective function

$$G(T, \beta) = \int_0^T g(x(\beta, t), \beta, t)dt \tag{3.26}$$

in which function g is assumed to be twice continuously differentiable and bounded, is [16]

$$(w^\top \mathcal{F}_{x'})' - w^\top \mathcal{F}_x = -g_x \tag{3.27}$$

where w is the adjoint variable.

Re-writing Eq. (3.16) as a first order ODE, in the form given by Eq. (3.25),

$$\mathcal{F}(x', x, t, \beta) \equiv \begin{pmatrix} {x'}_1 - x_2 \\ \\ RMS{x'}_2 - U_2 x_2 - U_1 x_1 - \hat{r} \end{pmatrix} = 0 \tag{3.28}$$

in which $x_1 = u$, $x_2 = u'$, $x = \begin{pmatrix} {x_1}^\top & {x_2}^\top \end{pmatrix}^\top$, $U_1 = RA_1S - RMS'' + RA_2S'$, and $U_2 = RA_2S - 2RMS'$; matrices $\mathcal{F}_{x'}$ and $\mathcal{F}_x$ are, respectively

$$\mathcal{F}_{x'} = \begin{pmatrix} I & 0 \\ 0 & RMS \end{pmatrix} \tag{3.29}$$

$$\mathcal{F}_x = \begin{pmatrix} 0 & -I \\ -U_1 & -U_2 \end{pmatrix} \tag{3.30}$$

Let the adjoint variable w in Eq. (3.27), corresponding to state vector x in Eq. (3.28), be $w = \begin{pmatrix} {w_1}^\top & {w_2}^\top \end{pmatrix}^\top$. Then, the adjoint equation that corresponds to Eq. (3.28), re-written in the form given by Eq. (3.27), is

$$\begin{aligned} &\left[\begin{pmatrix} {w_1}^\top & {w_2}^\top \end{pmatrix} \begin{pmatrix} I & 0 \\ 0 & RMS \end{pmatrix} \right]' \\ - &\begin{pmatrix} {w_1}^\top & {w_2}^\top \end{pmatrix} \begin{pmatrix} 0 & -I \\ -U_1 & -U_2 \end{pmatrix} = - \begin{pmatrix} g_{x_1} & g_{x_2} \end{pmatrix} \end{aligned} \tag{3.31}$$

which is written component-wise as

$${w_1}'^\top + {w_2}^\top (RA_1S - RMS'' + RA_2S') = -g_{x_1} \tag{3.32}$$

$${w_2}'^\top RMS + {w_2}^\top \Big((RMS)' + RA_2S - 2RMS' \Big) + {w_1}^\top = -g_{x_2} \tag{3.33}$$

By differentiating Eq. (3.33) with respect to time and subtracting the result from Eq. (3.32), the following differential equation is obtained:

$$\begin{aligned} {w_2}''^\top RMS \;&+\; {w_2}'^\top \Big(2(RM)'S + RA_2S \Big) \\ &+\; {w_2}^\top \Big((RA_2S - 2RMS')' + (RMS)'' - RA_1S \\ &+\; RMS'' - RA_2S' \Big) = g_{x_1} - {g_{x_2}}' \end{aligned} \tag{3.34}$$

in which the matrix coefficient of the ${w_2}^\top$ term,

$$U_0 = (RA_2S - 2RMS')' + (RMS)'' - RA_1S + RMS'' - RA_2S'$$

is re-written, by expanding and simplifying terms, as

$$U_0 = R'A_2S + RA_2'S + R''MS + RM''S + 2R'M'S - RA_1S$$

Replacing the matrix coefficient U_0 in Eq. (3.34) and transposing the equation, the following differential equation is obtained:

$$\begin{aligned} (RMS)^\top {w_2}'' &+ \Big(2S^\top {M'}^\top R^\top + 2S^\top M^\top {R'}^\top + S^\top {A_2}^\top R^\top\Big) {w_2}' \\ &+ \Big(S^\top {A_2}^\top {R'}^\top + S^\top {A_2'}^\top R^\top + S^\top M^\top {R''}^\top \\ &+ S^\top {M''}^\top R^\top + 2S^\top {M'}^\top {R'}^\top - S^\top {A_1}^\top R^\top\Big) w_2 \\ &= ({g_{x_1}} - {g_{x_2}}')^\top \end{aligned} \tag{3.35}$$

This is an ODE, according to Theorem 3.1, since matrix RMS is non-singular.

In order to re-write the linearized index-3 equation of motion defined by Eqs. (3.3) and (3.4) in the implicit first-order form of Eq. (3.25), consider vector v defined as $v = \begin{pmatrix} {v_1}^\top & {v_2}^\top & {v_3}^\top \end{pmatrix}^\top = \begin{pmatrix} q^\top & {q'}^\top & \lambda^\top \end{pmatrix}^\top$, in which $v_1 \equiv q$, $v_2 \equiv q'$, and $v_3 \equiv \lambda$. As a result, the linearized index-3 equation of motion of Eqs. (3.3) and (3.4) is re-written as

$$\mathcal{F}(v', v, t, \beta) \equiv \begin{pmatrix} {v_1}' - v_2 \\ Mv_2' - A_1v_1 - A_2v_2 - Bv_3 - s \\ Cv_1 + r \end{pmatrix} = 0 \tag{3.36}$$

Consequently, the adjoint equation of Eq. (3.36), obtained by applying the procedure defined in Eq. (3.27), is

$$\begin{aligned} & \left[\begin{pmatrix} {\mu_1}^\top & {\mu_2}^\top & {\mu_3}^\top \end{pmatrix} \begin{pmatrix} I & 0 & 0 \\ 0 & M & 0 \\ 0 & 0 & 0 \end{pmatrix} \right]' \\ - & \begin{pmatrix} {\mu_1}^\top & {\mu_2}^\top & {\mu_3}^\top \end{pmatrix} \begin{pmatrix} 0 & -I & 0 \\ -A_1 & -A_2 & -B \\ C & 0 & 0 \end{pmatrix} \\ = & -\begin{pmatrix} g_{v_1} & g_{v_2} & g_{v_3} \end{pmatrix} \end{aligned} \tag{3.37}$$

in which $\mu = \begin{pmatrix} {\mu_1}^\top & {\mu_2}^\top & {\mu_3}^\top \end{pmatrix}^\top$ is the adjoint variable. By expanding terms, Eq. (3.37) is re-written component-wise as

$${\mu_1}'^\top + {\mu_2}^\top A_1 - {\mu_3}^\top C = -g_{v_1} \tag{3.38}$$

$${\mu_2}'^\top M + {\mu_2}^\top M' + {\mu_1}^\top + {\mu_2}^\top A_2 = -g_{v_2} \tag{3.39}$$

$${\mu_2}^\top B = -g_{v_3} \tag{3.40}$$

By differentiating Eq. (3.39) with respect to time, subtracting the result from Eq. (3.38), applying the conjugate-transpose operator, and grouping terms; the index-3 adjoint DAE is obtained,

$$M^\top {\mu_2}'' + (2{M'}^\top + {A_2}^\top){\mu_2}' + ({M''}^\top + {A_2}'^\top - {A_1}^\top)\mu_2 + C^\top \mu_3 = \hat{s} \tag{3.41}$$

$$0 = B^\top \mu_2 + \hat{r} \tag{3.42}$$

where $\hat{s} \equiv {g_{v_1}}^\top - {g_{v_2}}'^\top$ and $\hat{r} \equiv {g_{v_3}}^\top$ are bounded, as a result of smoothness and

boundedness properties of function g. In order to obtain an underlying ODE for the DAE of Eqs. (3.41) and (3.42), the change of variable $w = S^\top \mu_2$ is performed, after pre-multiplying Eq. (3.41) with matrix $S^\top$. Since $0 = CS = S^\top C^\top$, the term containing the algebraic variable μ_3 vanishes, yielding the ODE

$$S^\top M^\top {\mu_2}'' + S^\top (2{M'}^\top + {A_2}^\top){\mu_2}' + S^\top ({M''}^\top + {A_2'}^\top - {A_1}^\top)\mu_2 = S^\top \hat{s} \tag{3.43}$$

Because of the identity of Eq. (3.10), the inverse of matrix $\begin{pmatrix} S & F \end{pmatrix}^\top$ is explicitly obtained as

$$\left[\begin{pmatrix} S & F \end{pmatrix}^\top\right]^{-1} = \begin{pmatrix} S^\top \\ F^\top \end{pmatrix}^{-1} = \begin{pmatrix} R \\ C \end{pmatrix}^\top = \begin{pmatrix} R^\top & C^\top \end{pmatrix} \tag{3.44}$$

Using the expression of Eq. (3.13) for matrix F, the matrix-vector product $\begin{pmatrix} S & F \end{pmatrix}^\top \mu_2$ is

$$\begin{pmatrix} S^\top \\ F^\top \end{pmatrix} \mu_2 = \begin{pmatrix} w \\ F^\top \mu_2 \end{pmatrix} = \begin{pmatrix} w \\ \left((CB)^{-1}\right)^\top B^\top \mu_2 \end{pmatrix} = \begin{pmatrix} w \\ -\left((CB)^{-1}\right)^\top \hat{r} \end{pmatrix} \tag{3.45}$$

As a result, μ_2 may be defined as a linear expression in variable w through the identity

$$\begin{aligned} \mu_2 &= \begin{pmatrix} S^\top \\ F^\top \end{pmatrix}^{-1} \begin{pmatrix} w \\ -\left((CB)^{-1}\right)^\top \hat{r} \end{pmatrix} \\ &= \begin{pmatrix} R^\top & C^\top \end{pmatrix} \begin{pmatrix} w \\ -\left((CB)^{-1}\right)^\top \hat{r} \end{pmatrix} = R^\top w - C^\top (B^\top C^\top)^{-1} \hat{r} \end{aligned} \tag{3.46}$$

By differentiating Eq. (3.46) once and twice with respect to time, expressions for first and second derivatives of μ_2 are obtained,

$${\mu_2}' = {R'}^\top w + R^\top w' - \left(C^\top (B^\top C^\top)^{-1} \hat{r}\right)' \tag{3.47}$$

$${\mu_2}'' = {R''}^\top w + 2{R'}^\top w' + R^\top w'' - \left(C^\top (B^\top C^\top)^{-1} \hat{r}\right)'' \tag{3.48}$$

As a result of replacing the adjoint variable μ_2 and its first and second derivatives with respect to time with the corresponding expressions of Eqs. (3.46), (3.47), and (3.48) in Eq. (3.43), the following underlying ODE is obtained for the index-3 adjoint DAE:

$$\begin{aligned}
(RMS)^\top w'' &+ \Big(2S^\top {M'}^\top R^\top + 2S^\top M^\top {R'}^\top + S^\top {A_2}^\top R^\top\Big) w' \\
&+ \Big(S^\top {A_2}^\top {R'}^\top + S^\top {A_2'}^\top R^\top + S^\top M^\top {R''}^\top + S^\top {M''}^\top R^\top \\
&+ 2S^\top {M'}^\top {R'}^\top - S^\top {A_1}^\top R^\top\Big) w \\
&= S^\top \hat{s} + S^\top M^\top \big(C^\top (B^\top C^\top)^{-1} \hat{r}\big)'' \qquad (3.49) \\
&+ (2S^\top {M'}^\top + S^\top {A_2}^\top)\big(C^\top (B^\top C^\top)^{-1} \hat{r}\big)' \\
&+ \Big(S^\top {M''}^\top + S^\top {A_2}^\top - S^\top {A_1}^\top\Big) C^\top (B^\top C^\top)^{-1} \hat{r}
\end{aligned}$$

The homogeneous parts of Eqs. (3.35) and (3.49) are identical. Therefore, the ODE of Eq. (3.49), which is the essential underlying ODE of the adjoint index-3 DAE of Eqs. (3.41) and (3.42) corresponding to the linearized index-3 DAE of motion, has the same stability properties as the ODE of Eq. (3.35). The ODE of Eq. (3.35) is the adjoint equation of the ODE of Eq. (3.16), which is the essential underlying ODE of the linearized index-3 DAE of motion. As a result, the following theorem establishes stability properties of the adjoint index-3 DAE that corresponds to the index-3 DAE of motion defined in Eqs. (3.1) and (3.2):

Theorem 3.3. *With the assumptions of theorem 3.1 regarding the mass matrix and constraint Jacobian, if the index-3 DAE of motion defined by Eqs.* (3.1) *and* (3.2) *is stable in the forward direction, then the corresponding adjoint index-3 DAE is stable*

in the backward direction.

Proof. The linearized index-3 DAE of motion of Eqs. (3.3) and (3.4) has locally the same solution as the original non-linear (quasi-linear) index-3 DAE of motion of Eqs. (3.1) and (3.2), which is assumed stable in the forward direction. Therefore, the linearized index-3 DAE of motion is also stable in the forward direction. Since the solution of the ODE of Eq. (3.16) is obtained from that of the linearized index-3 DAE of motion by the linear change of variable of Eq. (3.5), in which matrix R is bounded, it follows that the ODE of Eq. (3.16) is stable in the forward direction. Hence, the ODE of Eq. (3.28), which is the ODE of Eq. (3.16) written as a first order ODE, is stable in the forward direction.

The ODE of Eq. (3.35) is stable in the backward direction because [16] it is the adjoint equation of the ODE of Eq. (3.28), which is stable in the forward direction. Since, as shown above, the underlying ODE defined in Eq. (3.49) has the same homogeneous part as the ODE of Eq. (3.35), the ODE of Eq. (3.49) is also stable in the backward direction. The ODE of Eq. (3.49) is obtained from the adjoint index-3 DAE of Eqs. (3.41) and (3.42) by applying the linear change of coordinates of Eq. (3.46), in which matrices R, C, and B and vector $\hat{r}$ are bounded. Therefore, the adjoint index-3 DAE of Eqs. (3.41) and (3.42) has the same stability properties as the underlying ODE of Eq. (3.49). As a result, the adjoint index-3 DAE of Eqs. (3.41) and (3.42) is also stable in the backward direction.

It should be noted that the mass matrix is not assumed to be non-singular. The mass matrix may be singular, but positive-definite on the null-space of the constraint

Jacobian [52]. ■

3.3 The Adjoint Coordinate Partitioning Underlying ODE

Consider a partitioning $q = \begin{pmatrix} {u_{m\times 1}}^\top & {v_{d\times 1}}^\top \end{pmatrix}^\top$ of the generalized coordinate vector q into *dependent* part u and *independent* part v, such that the multibody system constraint Jacobian Φ_q is correspondingly partitioned as $\begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix}$ with Φ_u non-singular. Since the constraint Jacobian is assumed to have full row-rank at all times, such a partition always exists. In general, $\Phi_q = P_{row} \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} P_{col}$, in which $P_{row\,m\times m}$ and $P_{col\,n\times n}$ are permutation matrices, and such *foregoing* partitioning is only locally valid in an open neighborhood of a solution [58]. For simplicity, permutation matrix P_{row} is assumed to be the $m \times m$ identity matrix, and P_{col} is assumed to be the $n \times n$ identity matrix.

Since the index-3 adjoint DAE of Eqs. (3.41) and (3.42) has the constraint Jacobian $B^\top = C = -\Phi_q$, the index-3 adjoint DAE is re-written as

$$M\mu'' + D_1\mu' + D_2\mu + {\Phi_q}^\top \nu = \hat{s} \tag{3.50}$$

$$0 = \Phi_q \mu - \hat{r} \tag{3.51}$$

in which $\mu(t) = \begin{pmatrix} {\mu^u}^\top(t) & {\mu^v}^\top(t) \end{pmatrix}^\top$ is the adjoint differential variable $\mu_2(t)$ of Eq. (3.41), $\nu(t) \equiv -\mu_3(t)$ is the new adjoint algebraic variable, $D_1 = 2{M'}^\top + {A_2}^\top$, and $D_2 = {M''}^\top + {A_2'}^\top - {A_1}^\top$.

The index-3 adjoint DAE, in its new form of Eqs. (3.50) and (3.51), has the

same constraint Jacobian as the linearized DAE of motion of Eqs. (3.3) and (3.4). Therefore, the same foregoing partitioning, which was applied to the generalized coordinate vector q, is applied to the adjoint differential variable μ. As a result, the partitioned form of the index-3 adjoint DAE is

$$\begin{pmatrix} M^{uu} & M^{uv} \\ M^{vu} & M^{vv} \end{pmatrix} \begin{pmatrix} \mu^{u\prime\prime} \\ \mu^{v\prime\prime} \end{pmatrix} + \begin{pmatrix} {D_1}^{uu} & {D_1}^{uv} \\ {D_1}^{vu} & {D_1}^{vv} \end{pmatrix} \begin{pmatrix} \mu^{u\prime} \\ \mu^{v\prime} \end{pmatrix} + \begin{pmatrix} {D_2}^{uu} & {D_2}^{uv} \\ {D_2}^{vu} & {D_2}^{vv} \end{pmatrix} \begin{pmatrix} \mu^{u} \\ \mu^{v} \end{pmatrix} + \begin{pmatrix} {\Phi_u}^{\top} \\ {\Phi_v}^{\top} \end{pmatrix} \nu = \begin{pmatrix} \hat{s}^u \\ \hat{s}^v \end{pmatrix} \tag{3.52}$$

$$0 = \Phi_u \mu^u + \Phi_v \mu^v - \hat{r} \tag{3.53}$$

where the right-side term $\hat{s} = \begin{pmatrix} \hat{s}^u \\ \\ \hat{s}^v \end{pmatrix}$ is $\hat{s} \equiv {g_q}^{\top} - {g_{q'}}^{\top}$, the term $\hat{r}$ is $\hat{r} \equiv {g_\lambda}^{\top}$, and g is the objective function defined in Eq. (3.26).

As a result of the constraint equation of Eq. (3.53) and matrix Φ_u being non-singular, dependent partition μ^u can be expressed as a function of independent partition μ^v, as follows:

$$\begin{aligned} \mu^u &= -{\Phi_u}^{-1}\Phi_v \mu^v + {\Phi_u}^{-1} {g_y}^{\top} \\ &\equiv \Psi_{11}\mu^v + \psi_1 \end{aligned} \tag{3.54}$$

in which matrix Ψ_{11} and vector ψ_1 are

$$\Psi_{11} = -{\Phi_u}^{-1}\Phi_v \tag{3.55}$$

$$\psi_1 = {\Phi_u}^{-1} {g_y}^{\top} \tag{3.56}$$

Consider a $m \times n$ continuously differentiable matrix function $\Delta(q, \beta, t) =$

$\begin{pmatrix} \delta_1 \\ \vdots \\ \delta_m \end{pmatrix}$ with row vector functions $\delta_{i1\times n}(q,\beta,t),\ i=1,2,\ldots,m$. Then, for $n\times 1$ column vector functions $\rho(\beta,t)$ and $\gamma(\beta,t)$, not depending on q,

$$(\Delta\rho)_q\gamma = \left(\begin{pmatrix} \delta_1 \\ \vdots \\ \delta_m \end{pmatrix}\rho\right)_q\gamma = \begin{pmatrix} \delta_1\rho \\ \vdots \\ \delta_m\rho \end{pmatrix}_q\gamma = \begin{pmatrix} (\delta_1\rho)_q \\ \vdots \\ (\delta_m\rho)_q \end{pmatrix}\gamma \tag{3.57}$$

Since, for any $i=1,2,\ldots,m$, the gradient $(\delta_i\rho)_q$ is a row-vector,

$$(\Delta\rho)_q\gamma = \begin{pmatrix} (\delta_1\rho)_q\gamma \\ \vdots \\ (\delta_m\rho)_q\gamma \end{pmatrix} \tag{3.58}$$

For an arbitrary scalar function $\delta(q,\beta,t)$ and vectors ρ and γ that do not depend on q, the product $(\delta\rho)_q\gamma$ can be re-written as $(\rho^\top\delta^\top)_q\gamma$, since the product of row-vector δ with column-vector ρ is also a scalar. Vector ρ does not depend on q. Therefore,

$$(\rho^\top\delta^\top)_q\gamma = \rho^\top(\delta^\top)_q\gamma = \gamma^\top\left((\delta^\top)_q\right)^\top\rho \tag{3.59}$$

since $\rho^\top(\delta^\top)_q\gamma$ is a scalar. As a result,

$$(\Delta\rho)_q\gamma = \begin{pmatrix} \gamma^\top\left((\delta_1{}^\top)_q\right)^\top\rho \\ \vdots \\ \gamma^\top\left((\delta_m{}^\top)_q\right)^\top\rho \end{pmatrix} = \begin{pmatrix} \gamma^\top\left((\delta_1{}^\top)_q\right)^\top \\ \vdots \\ \gamma^\top\left((\delta_m{}^\top)_q\right)^\top \end{pmatrix}\rho \tag{3.60}$$

According to the partitioning $q = \begin{pmatrix} u_{m\times 1}{}^\top & v_{d\times 1}{}^\top \end{pmatrix}^\top$, matrix Δ and vector ρ

are re-written as $\Delta = \begin{pmatrix} \Delta^u & \Delta^v \end{pmatrix}$ and $\rho = \begin{pmatrix} \rho^{u\top} & \rho^{v\top} \end{pmatrix}^\top$, respectively. Therefore,

$$\begin{aligned}
&\left(\begin{pmatrix} \Delta^u & \Delta^v \end{pmatrix} \begin{pmatrix} \rho^u \\ \rho^v \end{pmatrix} \right)_q \gamma = (\Delta^u \rho^u + \Delta^v \rho^v)_q \gamma \\
&= \left(\begin{pmatrix} \begin{pmatrix} {\delta_1}^u \\ \vdots \\ {\delta_m}^u \end{pmatrix} \rho^u \end{pmatrix}_q \gamma + \begin{pmatrix} \begin{pmatrix} {\delta_1}^v \\ \vdots \\ {\delta_m}^v \end{pmatrix} \rho^v \end{pmatrix}_q \gamma \right. \\
&= \begin{pmatrix} \gamma^\top ((\delta_1{}^{u\top})_q)^\top \\ \vdots \\ \gamma^\top ((\delta_m{}^{u\top})_q)^\top \end{pmatrix} \rho^u + \begin{pmatrix} \gamma^\top ((\delta_1{}^{v\top})_q)^\top \\ \vdots \\ \gamma^\top ((\delta_m{}^{v\top})_q)^\top \end{pmatrix} \rho^v
\end{aligned} \tag{3.61}$$

Assume the constraint function vector $\Phi(q, \beta, t)$ has the form $\Phi(q, \beta, t) = \begin{pmatrix} \varphi_1(q, \beta, t) & \dots & \varphi_m(q, \beta, t) \end{pmatrix}^\top$ and consider $\Delta \equiv \Phi_q$, $\rho \equiv \mu$, and $\gamma \equiv q'$. As a result,

$$(\Phi_q \mu)_q q' = (\Delta \rho)_q \gamma \equiv \Theta \mu \tag{3.62}$$

in which

$$\Theta = \begin{pmatrix} \Theta^u & \Theta^v \end{pmatrix} = \begin{pmatrix} q'^\top ((\varphi_{1_q}{}^\top)_q)^\top \\ \vdots \\ q'^\top ((\varphi_{m_q}{}^\top)_q)^\top \end{pmatrix} \tag{3.63}$$

and

$$\Theta^u = \begin{pmatrix} q'^\top ((\varphi_{1_u}{}^\top)_q)^\top \\ \vdots \\ q'^\top ((\varphi_{m_u}{}^\top)_q)^\top \end{pmatrix} \tag{3.64}$$

$$\Theta^v = \begin{pmatrix} q'^{\top}\left(\left(\varphi_{1_v}{}^{\top}\right)_q\right)^{\top} \\ \vdots \\ q'^{\top}\left(\left(\varphi_{m_v}{}^{\top}\right)_q\right)^{\top} \end{pmatrix} \tag{3.65}$$

In order to obtain the first derivative of adjoint variable μ, Eq. (3.51) is differentiated once with respect to time,

$$\Phi_q \mu' + \Phi_{q,t}\mu + \left(\Phi_q \mu\right)_q q' = \frac{d}{dt}\left(g_y{}^{\top}\right) \tag{3.66}$$

With the notation defined in Eqs. (3.63) through (3.65), and the identity of Eq. (3.62), Eq. (3.66) is re-written as

$$\Phi_u \mu^{u\prime} + \Phi_v \mu^{v\prime} + \Phi_{u,t}\mu^u + \Phi_{v,t}\mu^v + \Theta^u \mu^u + \Theta^v \mu^v = \frac{d}{dt}\left(g_y{}^{\top}\right) \tag{3.67}$$

Therefore, the first derivative of the adjoint variable dependent partition is

$$\begin{aligned} \mu^{u\prime} &= -\Phi_u{}^{-1}\Phi_v \mu^{v\prime} - \Phi_u{}^{-1}\left(\Phi_{u,t} + \Theta^u\right)\mu^u \\ &- \Phi_u{}^{-1}\left(\Phi_{v,t} + \Theta^v\right)\mu^v + \Phi_u{}^{-1}\frac{d}{dt}\left(g_y{}^{\top}\right) \\ &\equiv \Psi_{22}\mu^{v\prime} + \Psi_{21}\mu^v + \psi_2 \end{aligned} \tag{3.68}$$

in which matrices Ψ_{22} and Ψ_{21} and vector ψ_2 are

$$\Psi_{22} = \Psi_{11} \tag{3.69}$$

$$\Psi_{21} = -\Phi_u{}^{-1}\left(-\left(\Phi_{u,t} + \Theta^u\right)\Phi_u{}^{-1}\Phi_v + \Phi_{v,t} + \Theta^v\right) \tag{3.70}$$

$$\psi_2 = -\Phi_u{}^{-1}\left(\Phi_{u,t} + \Theta^u\right)\psi_1 + \Phi_u{}^{-1}\frac{d}{dt}\left(g_y{}^{\top}\right) \tag{3.71}$$

Differentiating Eq.(3.66) with respect to time, the second order adjoint con-

straint equation is obtained as

$$\begin{aligned} \Phi_q\mu'' &+ 2\Phi_{q,t}\mu' + (\Phi_q\mu')_q q' + \Phi_{q,tt}\mu \\ &+ (\Phi_{q,t}\mu)_q q' + \frac{d}{dt}((\Phi_q\mu)_q q') = \frac{d^2}{dt^2}({g_y}^\top) \end{aligned} \tag{3.72}$$

in which

$$\frac{d}{dt}((\Phi_q\mu)_q q') = \frac{d}{dt}(\Theta\mu) = \Theta_t\mu + \Theta\mu' + (\Theta\mu)_q q' + (\Theta\mu)_{q'} q'' \tag{3.73}$$

$$(\Theta\mu)_{q'} = ((\Phi_q\mu)_q q')_{q'} = (\Phi_q\mu)_q \tag{3.74}$$

Applying the identity of Eq. (3.60) to $(\Theta\mu)_q q'$, in which $\Delta \equiv \Theta$, $\rho \equiv \mu$, and $\gamma \equiv q'$,

$$(\Theta\mu)_q q' = \begin{pmatrix} {q'}^\top(({\theta_1}^\top)_q)^\top \\ \vdots \\ {q'}^\top(({\theta_m}^\top)_q)^\top \end{pmatrix} \mu \equiv \Upsilon\mu \tag{3.75}$$

where θ_i, $i = 1, 2, \ldots, m$ are the rows of matrix $\Theta_{m\times n}$ and

$$\Upsilon \equiv \begin{pmatrix} {q'}^\top(({\theta_1}^\top)_q)^\top \\ \vdots \\ {q'}^\top(({\theta_m}^\top)_q)^\top \end{pmatrix} \tag{3.76}$$

Applying the partition $q = \begin{pmatrix} {u_{m\times 1}}^\top & {v_{d\times 1}}^\top \end{pmatrix}^\top$ to each row θ_i, $i = 1, 2, \ldots, m$, the $m \times n$ matrix Υ is partitioned as

$$\Upsilon = \begin{pmatrix} \Upsilon^u & \Upsilon^v \end{pmatrix} \tag{3.77}$$

in which

$$\Upsilon^{u}{}_{m\times m} \equiv \begin{pmatrix} {q'}^{\top}\left(\left({\theta_1}^{u\top}\right)_q\right)^{\top} \\ \vdots \\ {q'}^{\top}\left(\left({\theta_m}^{u\top}\right)_q\right)^{\top} \end{pmatrix} \tag{3.78}$$

and

$$\Upsilon^{v}{}_{m\times m} \equiv \begin{pmatrix} {q'}^{\top}\left(\left({\theta_1}^{v\top}\right)_q\right)^{\top} \\ \vdots \\ {q'}^{\top}\left(\left({\theta_m}^{v\top}\right)_q\right)^{\top} \end{pmatrix} \tag{3.79}$$

Applying the identity of Eq. (3.60) to terms $\left(\Phi_q\mu'\right)_q q'$, $\left(\Phi_{q,t}\mu\right)_q q'$, and $\left(\Theta\mu\right)_{q'} q'' = \left(\Phi_q\mu\right)_q q''$,

$$\left(\Phi_q\mu'\right)_q q' = \Theta\mu' \tag{3.80}$$

$$\left(\Phi_{q,t}\mu\right)_q q' = \begin{pmatrix} {q'}^{\top}\left(\left({\varphi_1}_{q,t}{}^{\top}\right)_q\right)^{\top} \\ \vdots \\ {q'}^{\top}\left(\left({\varphi_m}_{q,t}{}^{\top}\right)_q\right)^{\top} \end{pmatrix} \mu \equiv T\mu = \begin{pmatrix} T^u & T^v \end{pmatrix} \mu \tag{3.81}$$

$$T = \begin{pmatrix} {q'}^{\top}\left(\left({\varphi_1}_{q,t}{}^{\top}\right)_q\right)^{\top} \\ \vdots \\ {q'}^{\top}\left(\left({\varphi_m}_{q,t}{}^{\top}\right)_q\right)^{\top} \end{pmatrix} \tag{3.82}$$

$$T^u = \begin{pmatrix} {q'}^{\top}\left(\left({\varphi_1}_{u,t}{}^{\top}\right)_q\right)^{\top} \\ \vdots \\ {q'}^{\top}\left(\left({\varphi_m}_{u,t}{}^{\top}\right)_q\right)^{\top} \end{pmatrix} \tag{3.83}$$

$$T^v = \begin{pmatrix} q'^{\top}\left(\left({\varphi_{1_{v,t}}}^{\top}\right)_q\right)^{\top} \\ \vdots \\ q'^{\top}\left(\left({\varphi_{m_{v,t}}}^{\top}\right)_q\right)^{\top} \end{pmatrix} \tag{3.84}$$

$$\left(\Phi_q \mu\right)_q q'' = \begin{pmatrix} q''^{\top}\left(\left({\varphi_{1_q}}^{\top}\right)_q\right)^{\top} \\ \vdots \\ q''^{\top}\left(\left({\varphi_{m_q}}^{\top}\right)_q\right)^{\top} \end{pmatrix} \mu \equiv Z\mu = \begin{pmatrix} Z^u & Z^v \end{pmatrix} \mu \tag{3.85}$$

$$Z = \begin{pmatrix} q''^{\top}\left(\left({\varphi_{1_q}}^{\top}\right)_q\right)^{\top} \\ \vdots \\ q''^{\top}\left(\left({\varphi_{m_q}}^{\top}\right)_q\right)^{\top} \end{pmatrix} \tag{3.86}$$

$$Z^u = \begin{pmatrix} q''^{\top}\left(\left({\varphi_{1_u}}^{\top}\right)_q\right)^{\top} \\ \vdots \\ q''^{\top}\left(\left({\varphi_{m_u}}^{\top}\right)_q\right)^{\top} \end{pmatrix} \tag{3.87}$$

$$Z^v = \begin{pmatrix} q''^{\top}\left(\left({\varphi_{1_v}}^{\top}\right)_q\right)^{\top} \\ \vdots \\ q''^{\top}\left(\left({\varphi_{m_v}}^{\top}\right)_q\right)^{\top} \end{pmatrix} \tag{3.88}$$

After applying the partitions $q = \begin{pmatrix} u^{\top} & v^{\top} \end{pmatrix}^{\top}$ and $\mu = \begin{pmatrix} \mu^{u\top} & \mu^{v\top} \end{pmatrix}^{\top}$ to Eq. (3.72) and using identities of Eqs. (3.73) through (3.88), the second derivative

of the adjoint variable constraint equation is

$$\begin{aligned}
\Phi_u\mu^{u\prime\prime} &+ \Phi_v\mu^{v\prime\prime} + 2\Phi_{u,t}\mu^{u\prime} + 2\Phi_{v,t}\mu^{v\prime} \\
&+ 2\Theta^u\mu^{u\prime} + 2\Theta^v\mu^{v\prime} + \Phi_{u,tt}\mu^u + \Phi_{v,tt}\mu^v \\
&+ T^u\mu^u + T^v\mu^v + \Theta^u_t\mu^u + \Theta^v_t\mu^v \\
&+ \Upsilon^u\mu^u + \Upsilon^v\mu^v + Z^u\mu^u + Z^v\mu^v = \frac{d^2}{dt^2}({g_y}^\top)
\end{aligned} \tag{3.89}$$

As a result, $\mu^{u\prime\prime}$ can be expressed as a linear function of the independent adjoint variable μ^v and its first and second derivatives,

$$\begin{aligned}
\mu^{u\prime\prime} &= -{\Phi_u}^{-1}\Big(\Phi_v\mu^{v\prime\prime} + 2(\Phi_{u,t} + \Theta^u)\big(-{\Phi_u}^{-1}\Phi_v\mu^{v\prime} \\
&+ ({\Phi_u}^{-1}(\Phi_{u,t} + \Theta^u){\Phi_u}^{-1}\Phi_v - {\Phi_u}^{-1}(\Phi_{v,t} + \Theta^v))\mu^v \\
&+ {\Phi_u}^{-1}(-{\Phi_u}^{-1}{g_y}^\top + \frac{d}{dt}({g_y}^\top))\big) + 2(\Phi_{v,t} + \Theta^v)\mu^{v\prime} \\
&+ (\Phi_{u,tt} + T^u + \Theta^u_t + \Upsilon^u + Z^u)(-{\Phi_u}^{-1}\Phi_v\mu^v + {\Phi_u}^{-1}{g_y}^\top) \\
&+ (\Phi_{v,tt} + T^v + \Theta^v_t + \Upsilon^v + Z^v)\mu^v - \frac{d^2}{dt^2}({g_y}^\top)\Big) \\
&\equiv \Psi_{33}\mu^{v\prime\prime} + \Psi_{32}\mu^{v\prime} + \Psi_{31}\mu^v + \psi_3
\end{aligned} \tag{3.90}$$

in which matrices $\Psi_{3,i}$, $i = 1, 2, 3$ and vector ψ_3 are

$$\Psi_{33} = \Psi_{11} \tag{3.91}$$

$$\Psi_{32} = 2\Psi_{21} \tag{3.92}$$

$$\begin{aligned}
\Psi_{31} &= -{\Phi_u}^{-1}\Big(2(\Phi_{u,t} + \Theta^u)\Psi_{21} + (\Phi_{u,tt} + T^u + \Theta^u_t + \Upsilon^u + Z^u)\Psi_{11} \\
&+ (\Phi_{v,tt} + T^v + \Theta^v_t + \Upsilon^v + Z^v))
\end{aligned} \tag{3.93}$$

$$\begin{aligned}\psi_3 &= -{\Phi_u}^{-1}\Big(2(\Phi_{u,t}+\Theta^u)\psi_2 \\ &+ (\Phi_{u,tt}+T^u+\Theta^u_t+\Upsilon^u+Z^u)\psi_1-\frac{d^2}{dt^2}({g_y}^\top)\Big) \end{aligned} \tag{3.94}$$

Expanding the upper-block equation in Eq. (3.52),

$$M^{uu}\mu^{u\prime\prime}+M^{uv}\mu^{v\prime\prime}+{D_1}^{uu}\mu^{u\prime}+{D_1}^{uv}\mu^{v\prime}+{D_2}^{uu}\mu^u+{D_2}^{uv}\mu^v+{\Phi_u}^\top\nu=\hat{s}^u \tag{3.95}$$

the Lagrange multiplier ν is expressed as a function of $\mu^{u\prime\prime}$, $\mu^{v\prime\prime}$, $\mu^{u\prime}$, $\mu^{v\prime}$, μ^u, and μ^v,

$$\begin{aligned}\nu={\Phi_u}^{-1^\top}\hat{s}^u &- {\Phi_u}^{-1^\top}(M^{uu}\mu^{u\prime\prime}+M^{uv}\mu^{v\prime\prime} \\ &+ {D_1}^{uu}\mu^{u\prime}+{D_1}^{uv}\mu^{v\prime} \\ &+ {D_2}^{uu}\mu^u+{D_2}^{uv}\mu^v)\end{aligned} \tag{3.96}$$

Substituting for $\mu^{u\prime\prime}$ by expression of Eq. (3.90), for $\mu^{u\prime}$ by expression of Eq. (3.68), and for μ^u by expression of Eq. (3.54) in Eq. (3.96),

$$\begin{aligned}\nu &= {\Phi_u}^{-1^\top}(\hat{s}^u-M^{uu}(\Psi_{33}\mu^{v\prime\prime}+\Psi_{32}\mu^{v\prime}+\Psi_{31}\mu^v+\psi_3)-M^{uv}\mu^{v\prime\prime} \\ &- {D_1}^{uu}(\Psi_{22}\mu^{v\prime}+\Psi_{21}\mu^v+\psi_2)-{D_1}^{uv}\mu^{v\prime} \\ &- {D_2}^{uu}(\Psi_{11}\mu^v+\psi_1)-{D_2}^{uv}\mu^v)\end{aligned} \tag{3.97}$$

Collecting terms,

$$\nu=\Psi_{43}\mu^{v\prime\prime}+\Psi_{42}\mu^{v\prime}+\Psi_{41}\mu^v+\psi_4 \tag{3.98}$$

where

$$\Psi_{43} = -{\Phi_u}^{-1^\top}(M^{uu}\Psi_{33}+M^{uv}) \tag{3.99}$$

$$\Psi_{42} = -{\Phi_u}^{-1^\top}(M^{uu}\Psi_{32}+{D_1}^{uu}\Psi_{22}+{D_1}^{uv}) \tag{3.100}$$

$$\Psi_{41} = -{\Phi_u}^{-1^\top}(M^{uu}\Psi_{31}+{D_1}^{uu}\Psi_{21}+{D_2}^{uu}\Psi_{11}+{D_2}^{uv}) \tag{3.101}$$

$$\psi_4 = {\Phi_u}^{-1^\top}(\hat{s}^u-M^{uu}\psi_3-{D_1}^{uu}\psi_2-{D_2}^{uu}\psi_1) \tag{3.102}$$

Expanding the lower-block equation in Eq. (3.52),

$$M^{vu}\mu^{u\prime\prime} + M^{vv}\mu^{v\prime\prime} + {D_1}^{vu}\mu^{u\prime} + {D_1}^{vv}\mu^{v\prime} + {D_2}^{vu}\mu^{u} + {D_2}^{vv}\mu^{v} + {\Phi_v}^{\top}\nu = \hat{s}^v \quad (3.103)$$

Substituting for $\mu^{u\prime\prime}$ by expression of Eq. (3.90), for $\mu^{u\prime}$ by expression of Eq. (3.68), for μ^u by expression of Eq. (3.54), and for ν by expression of Eq. (3.98) in Eq. (3.103),

$$\begin{aligned} & M^{vu}(\Psi_{33}\mu^{v\prime\prime} + \Psi_{32}\mu^{v\prime} + \Psi_{31}\mu^{v} + \psi_3) + M^{vv}\mu^{v\prime\prime} \\ + \;& {D_1}^{vu}(\Psi_{22}\mu^{v\prime} + \Psi_{21}\mu^{v} + \psi_2) + {D_1}^{vv}\mu^{v\prime} \\ + \;& {D_2}^{vu}(\Psi_{11}\mu^{v} + \psi_1) + {D_2}^{vv}\mu^{v} \\ + \;& {\Phi_v}^{\top}(\Psi_{43}\mu^{v\prime\prime} + \Psi_{42}\mu^{v\prime} + \Psi_{41}\mu^{v} + \psi_4) = \hat{s}^v \end{aligned} \quad (3.104)$$

and collecting terms, the coordinate partitioning underlying ODE of the adjoint variable DAE is obtained,

$$B_1\mu^{v\prime\prime} + B_2\mu^{v\prime} + B_3\mu^{v} = b_4 \quad (3.105)$$

where $d \times d$ matrices B_i, $i = 1, 2, 3$, and $d \times 1$ vector b_4 are

$$\begin{aligned} B_1 &= M^{vu}\Psi_{33} + M^{vv} + {\Phi_v}^{\top}\Psi_{43} \\ B_2 &= M^{vu}\Psi_{32} + {D_1}^{vu}\Psi_{22} + {D_1}^{vv} + {\Phi_v}^{\top}\Psi_{42} \\ B_3 &= M^{vu}\Psi_{31} + {D_1}^{vu}\Psi_{21} + {D_2}^{vu}\Psi_{11} + {D_2}^{vv} + {\Phi_v}^{\top}\Psi_{41} \\ b_4 &= \hat{s}^v - M^{vu}\psi_3 - {D_1}^{vu}\psi_2 - {D_2}^{vu}\psi_1 - {\Phi_v}^{\top}\psi_4 \end{aligned}$$

After re-arranging and factoring terms, matrices B_i, $i = 1, 2, 3$ are re-written as

$$B_1 = {X_0}^{\top} M X_0 \quad (3.106)$$

$$B_2 = {X_0}^\top \Big(2M^u(-{\Phi_u}^{-1})K_{21} + D_1 \Big) X_0 \tag{3.107}$$

$$\begin{aligned} B_3 &= {X_0}^\top \Big(M^u(-{\Phi_u}^{-1})(2K_{21}^u(-{\Phi_u}^{-1})K_{21} + K_{31}) \\ &+ {D_1}^u(-{\Phi_u}^{-1})K_{21} + D_2 \Big) X_0 \end{aligned} \tag{3.108}$$

where

$$\begin{aligned} M^u &= \begin{pmatrix} M^{uu} \\ \\ M^{vu} \end{pmatrix} \\ {D_1}^u &= \begin{pmatrix} {D_1}^{uu} \\ \\ {D_1}^{vu} \end{pmatrix} \\ X_{0n\times d} &= \begin{pmatrix} -{\Phi_u}^{-1}\Phi_v \\ \\ I_d \end{pmatrix} \end{aligned}$$

$$\begin{aligned} K_{21} &= \begin{pmatrix} K_{21}^u & K_{21}^v \end{pmatrix} \equiv \begin{pmatrix} \Phi_{u,t} + \Theta^u & \Phi_{v,t} + \Theta^v \end{pmatrix} = \Phi_{q,t} + \Theta \\ K_{31} &\equiv \begin{pmatrix} \Phi_{u,tt} + T^u + \Theta_t^u + \Upsilon^u + Z^u & \Phi_{v,tt} + T^v + \Theta_t^v + \Upsilon^v + Z^v \end{pmatrix} \\ &= \Phi_{q,tt} + T + \Theta_t + \Upsilon + Z \end{aligned}$$

and I_d is the $d \times d$ identity matrix. In order for Eq. (3.105) to be a second order ODE, matrix B_1 must be nonsingular.

Corollary 3.4. *Matrix ${X_0}^\top M X_0$ is non-singular, where M is the mass matrix and* $X_0 = \begin{pmatrix} -{\Phi_u}^{-1}\Phi_v \\ \\ I_d \end{pmatrix}$.

Proof. Matrix $X_{d\times d} \equiv {X_0}^\top X_0 = ({\Phi_u}^{-1}\Phi_v)^\top {\Phi_u}^{-1}\Phi_v + I_d$ is positive definite because, for an arbitrary $d \times 1$ non-zero vector ξ, $\xi^\top X\xi = \|\xi\|^2 + \|{\Phi_u}^{-1}\Phi_v\xi\|^2 \geq \|\xi\|^2 > 0$. Moreover, $\Phi_q X_0 = \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} \begin{pmatrix} -{\Phi_u}^{-1}\Phi_v \\ I_d \end{pmatrix} = -\Phi_u{\Phi_u}^{-1}\Phi_v + \Phi_v = 0$, so

$$ {X_0}^\top {\Phi_q}^\top = 0 \tag{3.109} $$

Therefore, matrices ${X_0}^\top$ and X_0 have the properties of T_1 and T_2, respectively, from Theorem 3.1. Since ${X_0}^\top X_0$ is nonsingular, ${X_0}^\top B \equiv -{X_0}^\top {\Phi_q}^\top = 0$ and $CX_0 \equiv -\Phi_q X_0 = 0$ and the mass matrix is positive definite on the null-space of the constraint Jacobian. Therefore, matrix ${X_0}^\top M X_0$ is non-singular, according to Theorem 3.1 .

■

As a result, Eq. (3.105) represents the coordinate partitioning underlying ODE of the adjoint index-3 DAE of Eqs. (3.50) and (3.51). The coordinate partitioning [58] underlying ODE (CPUODE) can also be obtained through a procedure similar to the one used for obtaining the adjoint EUODE of Eq. (3.49), only using a different pre-multiplying matrix and different changes of variables. With the partitioning $\mu = \begin{pmatrix} {\mu^u}^\top & {\mu^v}^\top \end{pmatrix}^\top$, in which the dependent variable μ^u is replaced by the expression of Eq. (3.54), the adjoint variable μ is re-written as

$$ \mu = \begin{pmatrix} -{\Phi_u}^{-1}\Phi_v\mu^v + {\Phi_u}^{-1}{g_y}^\top \\ \mu^v \end{pmatrix} = X_0\mu^v + \begin{pmatrix} {\Phi_u}^{-1} \\ 0 \end{pmatrix} \hat{r} = X_0\mu^v + X_1\hat{r} \tag{3.110} $$

in which $X_1 = \begin{pmatrix} {\Phi_u}^{-1} \\ \\ 0 \end{pmatrix}$, $\hat{r} = {g_y}^{\top}$, and

$$\mu^v = \begin{pmatrix} 0 & I_d \end{pmatrix} \mu \tag{3.111}$$

Given an arbitrary non-zero vector $\xi = \begin{pmatrix} {\xi_1}^{\top} & {\xi_2}^{\top} \end{pmatrix}^{\top}$, consider now the product

$$ {X_0}^{\top}\xi = \begin{pmatrix} -({\Phi_u}^{-1}\Phi_v)^{\top} & I_d \end{pmatrix} \begin{pmatrix} \xi_1 \\ \\ \xi_2 \end{pmatrix} $$

A bound on the norm of ${X_0}^{\top}\xi$ is

$$\begin{aligned} \|{X_0}^{\top}\xi\| &= \| \begin{pmatrix} -({\Phi_u}^{-1}\Phi_v)^{\top} & I_d \end{pmatrix} \begin{pmatrix} \xi_1 \\ \\ \xi_2 \end{pmatrix} \| \\ &= \| -({\Phi_u}^{-1}\Phi_v)^{\top}\xi_1 + \xi_2\| \leq \|{\Phi_v}^{\top}\|\|{{\Phi_u}^{-1}}^{\top}\|\|\xi_1\| + \|\xi_2\| \end{aligned}$$

Since the constraint Jacobian is bounded, $\|\Phi_q\| \leq K_q$, matrix Φ_v is also bounded, $\|\Phi_v\| \leq \|\Phi_q\| \leq K_q$. Also, since matrix Φ_u is bounded and invertible, there is a positive constant K_u such that $\|{\Phi_u}^{-1}\| \leq K_u$. Hence,

$$\|{X_0}^{\top}\xi\| \leq K_q K_u \|\xi_1\| + \|\xi_2\| \leq K_q K_u \|\xi\| + \|\xi\| = (1 + K_q K_u)\|\xi\|$$

As a result, matrix ${X_0}^{\top}$ is bounded, with

$$\|{X_0}^{\top}\| \equiv \underset{\xi \neq 0}{sup} \frac{\|{X_0}^{\top}\xi\|}{\|\xi\|} \leq 1 + K_q K_u$$

Since constraint Jacobian Φ_q is twice continuously differentiable with respect to time, so is matrix ${X_0}^{\top}$. The rank of ${X_0}^{\top}$ is d, because I_d is the largest square block with a non-zero determinant. In addition, ${X_0}^{\top}B = {X_0}^{\top}{\Phi_q}^{\top} = 0$, as shown by

Eq. (3.109) in Corollary 3.4. Therefore, matrix ${X_0}^\top$ has the properties of Eqs. (3.6) through (3.9). As a result, matrix R can be selected as

$$R = {X_0}^\top \tag{3.112}$$

Let matrix $\tilde{S}$ be defined as

$$\tilde{S} = \begin{pmatrix} 0 & I_d \end{pmatrix}^\top = \begin{pmatrix} 0 \\ I_d \end{pmatrix} \tag{3.113}$$

Note that, although $R\tilde{S} = {X_0}^\top \begin{pmatrix} 0 \\ I_d \end{pmatrix} = I_d$, $C\tilde{S} = -\Phi_q \tilde{S} \neq 0$. Therefore, matrix $\tilde{S}$ is different from matrix S of Eqs. (3.11) and (3.12).

In order to obtain an underlying ODE of the adjoint index-3 DAE of Eqs. (3.50) and (3.51), through a procedure similar to that used for obtaining the EUODE of Eq. (3.16), the differential equation of Eq. (3.50) is pre-multiplied by $R = {X_0}^\top$. Accounting for the property that ${X_0}^\top {\Phi_q}^\top = 0$, the following differential equation is obtained:

$${X_0}^\top M\mu'' + {X_0}^\top D_1\mu' + {X_0}^\top D_2\mu = {X_0}^\top \hat{s} \tag{3.114}$$

where $D_1 = 2M' + {A_2}^\top$ and $D_2 = M'' + {A_2}'^\top - {A_1}^\top$. Then, the change of variable $\mu^v = \tilde{S}^\top \mu$ of Eq. (3.111) is performed, in which matrix $\tilde{S}$ is defined by Eq. (3.113). Differentiating Eq. (3.110) with respect to time once and twice, μ' and μ'' are obtained as functions of μ^v,$\mu^{v\prime}$, and $\mu^{v\prime\prime}$,

$$\mu' = X_0\mu^{v\prime} + {X_0}'\mu^v + (X_1\hat{r})' = X_0\mu^{v\prime} + X_2X_0\mu^v + (X_1\hat{r})' \tag{3.115}$$

$$\begin{aligned}\mu'' &= X_0\mu^{v\prime\prime} + 2{X_0}'\mu^{v\prime} + {X_0}''\mu^v + (X_1\hat{r})'' \\ &= X_0\mu^{v\prime\prime} + 2X_2X_0\mu^{v\prime} + X_3X_0\mu^v + (X_1\hat{r})'' \end{aligned} \tag{3.116}$$

where ${X_0}'$, ${X_0}''$, X_2, and X_3 are defined by Eqs. (A.55), (A.60), (A.56), and (A.61) in the Appendix. Replacing μ by the expression of Eq. (3.110); μ', and μ'' by expressions of Eqs. (3.115) and (3.116); replacing expressions for matrices D_1 and D_2 in Eq. (3.114); and factoring terms, an underlying ODE is obtained,

$$\begin{aligned} & {X_0}^\top M X_0\mu^{v\prime\prime} \\ + \; & {X_0}^\top(2MX_2 + 2M' + {A_2}^\top)X_0\mu^{v\prime} \\ + \; & {X_0}^\top(MX_3 + 2M'X_2 + {A_2}^\top X_2 + M'' + {{A_2}'}^\top - {A_1}^\top)X_0\mu^v \\ = \; & {X_0}^\top\hat{s} - {X_0}^\top M(X_1\hat{r})'' - {X_0}^\top D_1(X_1\hat{r})' - {X_0}^\top D_2X_1\hat{r} \end{aligned} \tag{3.117}$$

Equation (3.117) is an ODE, because ${X_0}^\top MX_0$ is nonsingular, according to Corollary 3.4.

Theorem 3.5. *If $R \equiv {X_0}^\top$, then the underlying ODE of Eq.* (3.117) *of the adjoint index-3 DAE of Eqs.* (3.50) *and* (3.51)*, obtained as a result of pre-multiplying the differential equation of Eq.* (3.50) *by R and performing the change of variable $\mu^v = \tilde{S}^\top\mu$, with $\tilde{S}$ defined in Eq.* (3.113)*, is the CPUODE of Eq.* (3.105)*.*

Proof.

The second derivative coefficient matrix of Eq. (3.117) is $U_1 \equiv {X_0}^\top MX_0$, which is equal to matrix B_1 of Eq. (3.106), which in turn is the second derivative coefficient matrix in Eq. (3.105). The first derivative coefficient matrix of Eq. (3.117)

is $U_2 \equiv {X_0}^\top(2MX_2 + 2M' + {A_2}^\top)X_0$. Replacing the expression for X_2, defined in Eq. (A.56) and using the identity of Eq. (A.55) in the Appendix, U_2 is re-written as

$$U_2 = {X_0}^\top(2M' + 2M \begin{pmatrix} -{\Phi_u}^{-1}\frac{d}{dt}(\Phi_q) \\ 0 \end{pmatrix} + {A_2}^\top)X_0$$

Using the identity of Eq. (A.46) in the Appendix, according to which $K_{21} = \frac{d}{dt}(\Phi_q)$, and replacing the expression $D_1 = 2{M'}^\top + {A_2}^\top$ in Eq. (3.107), the first derivative coefficient matrix B_2 in Eq. (3.105), is

$$\begin{aligned} B_2 &= {X_0}^\top\Big(2M^u(-{\Phi_u}^{-1})\frac{d}{dt}(\Phi_q) + 2M' + {A_2}^\top\Big)X_0 \\ &= {X_0}^\top\Big(2M' + 2\begin{pmatrix} M^u & M^v \end{pmatrix}\begin{pmatrix} -{\Phi_u}^{-1}\frac{d}{dt}(\Phi_q) \\ 0 \end{pmatrix} + {A_2}^\top\Big)X_0 \\ &= {X_0}^\top\Big(2M' + 2M\begin{pmatrix} -{\Phi_u}^{-1}\frac{d}{dt}(\Phi_q) \\ 0 \end{pmatrix} + {A_2}^\top\Big)X_0 = U_2 \end{aligned}$$

Therefore, first derivative coefficient matrices U_2 and B_2 of Eqs. (3.117) and (3.105) , respectively, are equal.

The coefficient matrix of the undifferentiated term in Eq. (3.117) is

$$U_3 \equiv {X_0}^\top(MX_3 + 2M'X_2 + {A_2}^\top X_2 + M'' + {A_2}'^\top - {A_1}^\top)X_0$$

Replacing expressions for X_2 and X_3; using the identities of Eqs. (A.46) and (A.47) and the expression for ${X_0}^{\top''}$ defined in Eq. (A.60) in the Appendix, matrix U_3 is

re-written as a sum of three matrices $U_3 = U_{31} + U_{32} + U_{33}$, where

$$
\begin{aligned}
U_{31} &\equiv {X_0}^{\top} M X_3 X_0 \\
&= {X_0}^{\top} M {X_0}''^{\top} \\
&= {X_0}^{\top} \begin{pmatrix} M^u & M^v \end{pmatrix} \begin{pmatrix} (-2{\Phi_u}^{-1})\frac{d}{dt}(\Phi_q) \begin{pmatrix} -{\Phi_u}^{-1}\frac{d}{dt}(\Phi_q) \\ 0 \end{pmatrix} - {\Phi_u}^{-1}\frac{d^2}{dt^2}(\Phi_q) \\ 0 \end{pmatrix} X_0 \\
&= {X_0}^{\top}(M^u(-2{\Phi_u}^{-1})\frac{d}{dt}(\Phi_q) \begin{pmatrix} -{\Phi_u}^{-1}\frac{d}{dt}(\Phi_q) \\ 0 \end{pmatrix} + M^u(-{\Phi_u}^{-1}\frac{d^2}{dt^2}(\Phi_q)))X_0 \\
&= {X_0}^{\top}(M^u(-{\Phi_u}^{-1})(2(\frac{d}{dt}(\Phi_q))^{u}(-{\Phi_u}^{-1})\frac{d}{dt}(\Phi_q) + \frac{d^2}{dt^2}(\Phi_q)))X_0 \\
&= {X_0}^{\top}(M^u(-{\Phi_u}^{-1})(2K_{21}^u(-{\Phi_u}^{-1})K_{21} + K_{31}))X_0 \\
U_{32} &\equiv 2{X_0}^{\top} M' X_2 X_0 + {X_0}^{\top} {A_2}^{\top} X_2 X_0 \\
&= {X_0}^{\top}(2M' + {A_2}^{\top}){X_0}' \\
&= {X_0}^{\top} \begin{pmatrix} (2M' + {A_2}^{\top})^u & (2M' + {A_2}^{\top})^v \end{pmatrix} \begin{pmatrix} -{\Phi_u}^{-1}\frac{d}{dt}(\Phi_q) \\ 0 \end{pmatrix} X_0 \\
&= {X_0}^{\top}(2M' + {A_2}^{\top})^u(-{\Phi_u}^{-1}K_{21})X_0 \\
U_{33} &\equiv {X_0}^{\top} M'' X_0 + {X_0}^{\top} {A_2}'^{\top} X_0 - {X_0}^{\top} {A_1}^{\top} X_0 \\
&= {X_0}^{\top}(M'' + {A_2}'^{\top} - {A_1}^{\top})X_0
\end{aligned}
$$

The matrix B_3 defined in Eq. (3.108) is also written as a sum of three matrices,

$B_3 = B_{31} + B_{32} + B_{33}$, where

$$\begin{aligned} B_{31} &\equiv X_0{}^\top (M^u(-\Phi_u{}^{-1})(2K_{21}^u(-\Phi_u{}^{-1})K_{21} + K_{31})X_0 \\ B_{32} &\equiv X_0{}^\top (D_1{}^u(-\Phi_u{}^{-1})K_{21})X_0 \\ B_{33} &\equiv X_0{}^\top D_2 X_0 \end{aligned}$$

After replacing expressions for $D_1 = 2M'^\top + A_2{}^\top$ and $D_2 = M''^\top + A_2{}'^\top - A_1{}^\top$, matrices $U_{3,1}$ and $B_{3,1}$, $U_{3,2}$ and $B_{3,2}$, and $U_{3,3}$ and $B_{3,3}$, respectively, are equal. Therefore, $U_3 = B_3$, hence the homogeneous parts of the underlying ODE of Eq. (3.117) and the CPUODE of Eq. (3.105) are identical. As a result, the two ordinary differential equations have the same stability properties.

Substituting for ψ_4 by the expression in Eq. (3.102) and factoring matrix $X_0{}^\top$, the right-side b_4 of Eq. (3.105) is re-written as

$$\begin{aligned} b_4 &= X_0{}^\top \hat{s} - (M^{vu} - \Phi_v{}^\top \Phi_u{}^{-1^\top})\psi_3 - (D_1{}^{vu} - \Phi_v{}^\top \Phi_u{}^{-1^\top})\psi_2 \\ &- (D_2{}^{vu} - \Phi_v{}^\top \Phi_u{}^{-1^\top})\psi_1 \\ &= X_0{}^\top \hat{s} - X_0{}^\top M^u \psi_3 - X_0{}^\top D_1{}^u \psi_2 - X_0{}^\top D_2{}^u \psi_1 \end{aligned} \tag{3.118}$$

After replacing expressions for ψ_i, $i = 1, 2, 3$, and using expressions for matrices K_{21} and K_{31} defined by Eqs. (A.46) and (A.47) in the Appendix, b_4 is written as a sum of three matrices, $b_4 = b_{41} + b_{42} + b_{43}$, where

$$\begin{aligned} b_{41} &= X_0{}^\top \hat{s} - X_0{}^\top M^u(-\Phi_u{}^{-1})\Big(2(\Phi_q{}')^u\big(-\Phi_u{}^{-1}(\Phi_{u,t} + \Theta^u)\Phi_u{}^{-1}\hat{r} \\ &+ \Phi_u{}^{-1}(\hat{r})'\big) + \big((\Phi_q{}'')^u\big)\Phi_u{}^{-1}\hat{r} - (\hat{r})''\Big) \end{aligned}$$

$$b_{42} = -X_0{}^\top D_1{}^u \Phi_u{}^{-1}\Big(-(\Phi_{u,t} + \Theta^u)\Phi_u{}^{-1}\hat{r} + (\hat{r})'\Big)$$

$$b_{43} = -{X_0}^{\top} {D_2}^{u} {\Phi_u}^{-1} \hat{r}$$

Note that for an arbitrary $n \times n$ matrix $A = \begin{pmatrix} A^u & A^v \end{pmatrix}$, in which block A^u is a $n \times m$ matrix and block A^v is a $n \times d$ matrix, the product $A^u {\Phi_u}^{-1}$ can be written as $A^u {\Phi_u}^{-1} = \begin{pmatrix} A^u & A^v \end{pmatrix} \begin{pmatrix} {\Phi_u}^{-1} \\ 0 \end{pmatrix} = AX_1$, in which matrix X_1 is defined by Eq. (A.51) in the Appendix. As a result, vectors $b_{4,i}$, $i = 1, 2, 3$, are

$$\begin{aligned} b_{41} &= {X_0}^{\top} \hat{s} + {X_0}^{\top} M X_1 \Big(2({\Phi_q}')^u \big(- {\Phi_u}^{-1} ({\Phi_q}')^u {\Phi_u}^{-1} \hat{r} \\ &+ {\Phi_u}^{-1} (\hat{r})' \big) + \big(({\Phi_q}'')^u \big) {\Phi_u}^{-1} \hat{r} - (\hat{r})'' \Big) \end{aligned} \tag{3.119}$$

$$b_{42} = -{X_0}^{\top} D_1 X_1 \Big(- (\Phi_{u,t} + \Theta^u) {\Phi_u}^{-1} \hat{r} + (\hat{r})' \Big) \tag{3.120}$$

$$b_{43} = -{X_0}^{\top} D_2 X_1 \hat{r} \tag{3.121}$$

After arranging terms by time derivative order of vector $\hat{r}$, b_{41} is re-written as

$$\begin{aligned} b_{41} &= {X_0}^{\top} \hat{s} + {X_0}^{\top} M X_1 \Big(\big((({\Phi_q}'')^u) {\Phi_u}^{-1} - 2({\Phi_q}')^u {\Phi_u}^{-1} ({\Phi_q}')^u {\Phi_u}^{-1} \big) \hat{r} \\ &+ 2({\Phi_q}')^u {\Phi_u}^{-1} (\hat{r})' - (\hat{r})'' \Big) \\ &= {X_0}^{\top} \hat{s} + {X_0}^{\top} M X_1 \Big(\big({\Phi_q}'' X_1 - 2{\Phi_q}' X_1 {\Phi_q}' X_1 \big) \hat{r} \\ &+ 2{\Phi_q}' X_1 (\hat{r})' - (\hat{r})'' \Big) \end{aligned} \tag{3.122}$$

The right-side of Eq. (3.117) is

$$\begin{aligned} u_4 &\equiv {X_0}^{\top} \hat{s} - {X_0}^{\top} M (X_1 \hat{r})'' - {X_0}^{\top} D_1 (X_1 \hat{r})' - {X_0}^{\top} D_2 X_1 \hat{r} \\ &= u_{41} + u_{42} + u_{43} \end{aligned}$$

in which

$$u_{41} = {X_0}^\top \hat{s} - {X_0}^\top M(X_1\hat{r})'' \tag{3.123}$$

$$u_{42} = -{X_0}^\top D_1(X_1\hat{r})' \tag{3.124}$$

$$u_{43} = -{X_0}^\top D_2 X_1\hat{r} \tag{3.125}$$

Vectors u_{43} and b_{43} are equal. Expanding the expression for vector u_{42} in Eq. (3.124),

$$u_{42} = -{X_0}^\top D_1 {X_1}'\hat{r} - {X_0}^\top D_1 X_1\hat{r}'$$

replacing the derivative of matrix X_1, defined by Eq. (A.66) in the Appendix, and using the expression for the first derivative of the Jacobian, ${\Phi_q}'$ defined in Eq. (A.46) in the Appendix, vector u_{42} is re-written as

$$\begin{aligned} u_{42} &= -{X_0}^\top D_1(-X_1{\Phi_q}'X_1)\hat{r} - {X_0}^\top D_1 X_1\hat{r}' \\ &= -{X_0}^\top D_1 X_1(-{\Phi_q}'X_1\hat{r} + \hat{r}') \\ &= -{X_0}^\top D_1 X_1(-({\Phi_q}')^u {\Phi_u}^{-1}\hat{r} + \hat{r}') \\ &= -{X_0}^\top D_1 X_1(-(\Phi_{u,t} + \Theta^u){\Phi_u}^{-1}\hat{r} + \hat{r}') \end{aligned}$$

which is equal to vector b_{42} of Eq. (3.120). Vector u_{41} is similarly expanded, using expressions for matrix X_1 and its derivatives of Eqs. (A.66) and (A.69) in the

Appendix, as

$$\begin{aligned}
u_{41} &= X_0{}^\top \hat{s} - X_0{}^\top M(X_1{}''\hat{r} + 2X_1{}'\hat{r}' + X_1\hat{r}'') \\
&= X_0{}^\top \hat{s} - X_0{}^\top M\Big((-X_1\Phi_q{}''X_1 + 2X_1\Phi_q{}'X_1\Phi_q{}'X_1)\hat{r} \\
&- 2X_1\Phi_q{}'X_1\hat{r}' + X_1\hat{r}''\Big) \\
&= X_0{}^\top \hat{s} - X_0{}^\top MX_1\Big(- (\Phi_q{}'')^u{\Phi_u}^{-1} + 2(\Phi_q{}')^u{\Phi_u}^{-1}(\Phi_q{}')^u{\Phi_u}^{-1}\Big)\hat{r} \\
&- 2(\Phi_q{}')^u{\Phi_u}^{-1}\hat{r}' + \hat{r}'' \\
&= X_0{}^\top \hat{s} + X_0{}^\top MX_1\Big(\big(\Phi_q{}''X_1 - 2\Phi_q{}'X_1\Phi_q{}'X_1\big)\hat{r} \\
&+ 2\Phi_q{}'X_1\hat{r}' - \hat{r}''\Big)
\end{aligned}$$

which is equal to vector b_{41} of Eq. (3.122). Since $u_{4,i} = b_{4,i}, i = 1, 2, 3$, the right-sides of Eqs. (3.117) and (3.105) are equal. As a result, Eqs. (3.117) and (3.105) are identical. ■

Stability of the CPUODE of Eq. (3.105) is shown by proving that its homogeneous ODE is the same as the homogeneous adjoint differential equation of a certain underlying ODE of the DAE of motion. In order to obtain this underlying ODE of the linearized index-3 DAE of Eqs. (3.3) and (3.4), the change of variable $w = S^\top q$ is performed, instead of the EUODE change of variable $u = Rq$, defined by Ascher et. al. [5]. Therefore, using an identity similar to the identity of Eq. (3.45),

$$\begin{aligned}
\begin{pmatrix} S^\top \\ F^\top \end{pmatrix} q = \begin{pmatrix} w \\ F^\top q \end{pmatrix} &= \begin{pmatrix} w \\ \big((CB)^{-1}\big)^\top B^\top q \end{pmatrix} \\
= \begin{pmatrix} w \\ \big((CB)^{-1}\big)^\top Cx \end{pmatrix} &= \begin{pmatrix} w \\ -\big((CB)^{-1}\big)^\top r \end{pmatrix}
\end{aligned} \tag{3.126}$$

and since $\left[\begin{pmatrix} S & F \end{pmatrix}^{\top}\right]^{-1} = \begin{pmatrix} R^{\top} & C^{\top} \end{pmatrix}$, as shown in Eq. (3.44), q is a linear function of w,

$$q = \begin{pmatrix} R^{\top} & C^{\top} \end{pmatrix} \begin{pmatrix} w \\ -((CB)^{-1})^{\top} r \end{pmatrix} = R^{\top} w - C^{\top}\big((CB)^{-1}\big)^{\top} r \tag{3.127}$$

Differentiating Eq. (3.127) twice with respect to time,

$$q'' = R^{\top} w'' + \sum_{j=0}^{1} \begin{pmatrix} 2 \\ j \end{pmatrix} (R^{\top})^{(2-j)} w^{(j)} - \Big(C^{\top}\big((CB)^{-1}\big)^{\top} r\Big)'' \tag{3.128}$$

Pre-multiplying the result by matrix RM, the following identity is obtained:

$$RMq'' = RMR^{\top} w'' + \sum_{j=0}^{1} \begin{pmatrix} 2 \\ j \end{pmatrix} RM(R^{\top})^{(2-j)} w^{(j)} - RM\Big(C^{\top}\big((CB)^{-1}\big)^{\top} r\Big)'' \tag{3.129}$$

Pre-multiplying Eq. (3.3) by matrix R, accounting for the property that $RB = 0$, and subtracting the result from the identity of Eq. (3.129), the following underlying ODE of the linearized index-3 DAE is obtained:

$$\begin{aligned} RMR^{\top} w'' &= (RA_2R^{\top} - 2RMR'^{\top})w' \\ &+ (RA_1R^{\top} + RA_2R'^{\top} - RMR''^{\top})w \\ &+ Rq + RM\big(C^{\top}((CB)^{-1})^{\top} r\big)'' \end{aligned} \tag{3.130}$$

This is an ODE, because the second derivative coefficient matrix is nonsingular, according to Corollary 3.4. Note that the underlying ODE of Eq. (3.130) is of similar form with the EUODE of Eq. (3.16), in which matrix S is replaced by matrix $R^{\top}$.

The adjoint differential equation corresponding to the underlying ODE of Eq. (3.130) is similarly obtained as the adjoint differential equation of Eq. (3.35). Using

a formulation analogous to that of Eqs. (3.27) through (3.34), in which matrix S is replaced by matrix $R^\top$ and matrix $S^\top$ is replaced by matrix R, the adjoint equation of the underlying ODE of Eq. (3.130) is obtained as

$$\begin{aligned}(RMR^\top)v'' &+ \Big(2RM'R^\top + 2RMR'^\top + RA_2{}^\top R^\top\Big)v' \\ &+ \Big(RA_2{}^\top R'^\top + RA_2{}'^\top R^\top + RMR''^\top + RM''R^\top \\ &+ 2RM'R'^\top - RA_1{}^\top R^\top\Big)v = (g_{q_1} - g_{q_2}{}')^\top\end{aligned} \tag{3.131}$$

By choosing matrix R as $R = X_0{}^\top$, using the expressions of matrix X_0 first and second derivatives defined by Eqs.(A.55) and (A.60), respectively, in the Appendix, and factoring terms, the homogeneous differential equation corresponding to Eq. (3.131) is

$$\begin{aligned}X_0{}^\top MX_0v'' &+ X_0{}^\top(2MX_2 + 2M' + A_2{}^\top)X_0v' \\ &+ X_0{}^\top(MX_3 + 2M'X_2 + A_2{}^\top X_2 \\ &+ M'' + A_2{}'^\top - A_1{}^\top)X_0v = 0\end{aligned} \tag{3.132}$$

which is the same as the homogeneous ODE corresponding to the CPUODE of Eq. (3.117). Since the index-3 DAE of motion is stable in the forward direction, the linearized index-3 DAE of motion is also stable in the forward direction. Therefore, its underlying ODE of Eq. (3.130) is stable in the forward direction, since the solution of the underlying ODE, $w = S^\top q$, is a linear function of the DAE solution q, in which matrix S is bounded. Then, the adjoint ODE of Eq. (3.131) is stable in the backward direction [16]. Therefore, the CPUODE of Eq. (3.105), whose homogeneous part

coincides with that of the adjoint ODE of Eq. (3.131), is stable in the backward direction. As a result, the theorem that follows has been proved.

Theorem 3.6. *The CPUODE of Eq.* (3.105) *of the adjoint index-3 DAE of Eqs.*(3.50) *and* (3.51) *is stable in the backward direction, provided that the multibody system index-3 DAE of motion of Eqs.* (3.1) *and* (3.2) *is stable in the forward direction.*

■

CHAPTER 4
IMPLICIT INTEGRATION OF MULTIBODY SYSTEM DIFFERENTIAL ALGEBRAIC EQUATIONS

The differential-algebraic equations of motion and the adjoint differential-algebraic equations of multibody dynamics are index-3 DAE. The DAE that result from the direct differentiation formulation [26] of design sensitivity analysis of multibody systems are also index-3 DAE. Differential-algebraic equations of index-3 must have a special structure [13]; i.e., they must be in Hessenberg form, in order for numerical integration methods to converge when they are directly applied to such higher-index DAE. The equations of motion, adjoint equations, and direct differentiation equations for a multibody system in which orientation of bodies is expressed using Euler parameters [27], cannot be brought in Hessenberg form, because the highest derivative coefficient matrices are singular [26, 52]. Direct discretization of an index-3 DAE introduces numerical stability difficulties; e.g., a BDF integration method applied to an index-3 DAE requires constant step-size for convergence [13]. As a result, the original index-3 DAE must be transformed into an equivalent lower index DAE or ODE; i.e., a DAE or ODE that has a numerical solution from which the solution of the original formulation can be recovered through linear changes of variables, of the form of Eq. (3.14) in Chapter 3, or by retaining only certain components of the lower index DAE solution. A lower index DAE formulation is obtained through index reduction methods [13]. Such methods are used to transform the original index-3 DAE into a lower index equivalent DAE (typically index-1, for multibody

systems [22]) or an underlying ODE [58]. After each integration step, the solution of the original system is recovered from the solution of the lower index system.

In some DAE formulations it may be possible [25] that not all the constraints can be enforced simultaneously. Appending the velocity and acceleration constraints, obtained by differentiating the multibody system position constraint once and twice, respectively, may result in an overdetermined system. Therefore, there are DAE formulations; e.g., index-2 equivalent formulations of index-3 DAE [25] that can only enforce a subset of the position, velocity, and acceleration constraints. As a result, the missing constraints require projection [25, 35] of the generalized coordinate vector, or its first or second derivatives, onto the corresponding constraint manifold [25].

In this chapter an index reduction method is selected for solving the equations of motion, the adjoint DAE, and the direct differentiation DAE, for a non-centroidal formulation of multibody system equations of motion. Equivalent index-1 formulations are presented for the three DAE systems, an implicit numerical integration package is chosen for solving the DAE, and reasons for the selected formulation and numerical integration method are explained.

4.1 Numerical Integration of Index-1 Differential-Algebraic Equations Using Backward Differentiation Formulas

Since numerical integration of DAE has similar stability properties to the numerical integration of stiff ODE [6] and because multibody systems can include stiff elements [41], implicit numerical integration methods; e.g., BDF methods are often used for integration of the multibody system DAE. In this thesis, the Implicit Differential-Algebraic solver (IDA) [31], is used for numerical integration of a multibody system DAE of motion index-1 equivalent formulation. Developed at Lawrence Livermore National Laboratory (LLNL) for solving ODE and index-1 DAE, IDA is a component of the LLNL's Suite of Non-linear Differential-Algebraic Solvers (SUNDIALS) package [30]. The implicit numerical integration method that is used in IDA is a BDF method of variable step-size and variable order up to five. In addition to its robustness, the IDA package has the advantages of implementing a complex step-size and order control algorithm based on the fixed leading coefficient method [13] and not requiring the user to start the integration; i.e., to provide the first s steps. In general, the first s steps of an s order multi-step integration method are supplied by the user, who must use a one-step integration method in the beginning and perform step-size and order control. This is internally done by the IDA integrator and is transparent to the user.

Consider the general form of an implicit index-0 ODE or index-1 DAE initial

value problem,

$$F(y, y', t) = 0 \tag{4.1}$$

together with consistent initial conditions

$$y(t_0) = y_0 \tag{4.2}$$

$$y'(t_0) = y'_0 \tag{4.3}$$

where y is the state vector and y' its derivative. In the fixed leading coefficient BDF method of order s, calculation of the state vector $y(t_{k+1})$ at time-step t_{k+1} proceeds as follows [13]:

1. Using approximations of the state vector y_{k-i}, at time-steps t_{k-i}, $i = 0, 1, \ldots, s$, a solution at time-step t_{k+1} is predicted by evaluating the predictor polynomial and its derivative at t_{k+1}. The predictor polynomial $\omega^P_{k+1}(t)$ interpolates y_{k-i} at time-steps t_{k-i}, $i = 0, 1, \ldots, s$; i.e.,

$$\omega^P_{k+1}(t_{k-i}) = y_{k-i}, i = 0, 1, \ldots, s \tag{4.4}$$

The predicted values of y and y' at t_{k+1} are, respectively

$$y^{(0)}_{k+1} = \omega^P_{k+1}(t_{k+1}) \tag{4.5}$$

$$y'^{(0)}_{k+1} = \omega'^P_{k+1}(t_{k+1}) \tag{4.6}$$

2. The corrector polynomial $\omega^C_{k+1}(t)$ is constructed such that it satisfies the DAE at t_{k+1}, and interpolates the predictor polynomial $\omega^P_{k+1}(t)$ at s equally spaced

points before t_{k+1},

$$\omega_{k+1}^{C}(t_{k+1} - ih_{k+1}) = \omega_{k+1}^{P}(t_{k+1} - ih_{k+1}), i = 1, 2, \ldots, s \tag{4.7}$$

$$F(\omega_{k+1}^{C}(t_{k+1}), \omega'^{C}_{k+1}(t_{k+1}), t_{k+1}) = 0 \tag{4.8}$$

where $h_{k+1} = t_{k+1} - t_k$.

3. The corrected solution $y_{k+1} = \omega_{k+1}^{C}(t_{k+1})$, implicitly defined by conditions of Eqs. (4.7) and (4.8), is obtained by solving the nonlinear equation [13]

$$F(y_{k+1}, y'^{(0)}_{k+1} - \frac{\alpha_s}{h_{k+1}}(y_{k+1} - y_{k+1}^{(0)}), t_{k+1}) = 0 \tag{4.9}$$

with the unknown y_{k+1}, where $\alpha_s = -\sum_{j=1}^{s} \frac{1}{j}$ and y'_{k+1} is given by the BDF integration method as

$$y'_{k+1} = y'^{(0)}_{k+1} - \frac{\alpha_s}{h_{k+1}}(y_{k+1} - y_{k+1}^{(0)}) \tag{4.10}$$

The solution of the nonlinear system of Eq. (4.9) is obtained by applying a modified Newton iteration,

$$y^{(\rho+1)} = y^{(\rho)} - cJ^{-1}F(y^{(\rho)}, \alpha_1 y^{(\rho)} + \alpha_2, t_{k+1}) \tag{4.11}$$

where

$$\alpha_1 = -\frac{\alpha_s}{h_{k+1}} \tag{4.12}$$

$$\alpha_2 = y'^{(0)}_{k+1} - \alpha_1 y_{k+1}^{(0)} \tag{4.13}$$

c is a scalar constant chosen to accelerate the rate of convergence of Newton iterations [13], and

$$J = F_y + \alpha_1 F_{y'} \tag{4.14}$$

is the Jacobian of nonlinear equation of Eq. (4.9).

The steps in the procedure presented above are transparent to the user of the integrator. The only function evaluators that the user must provide to the IDA integrator are [31] an evaluator of function $F(y, y', t)$ of Eq. (4.1) and an evaluator of Jacobian J of Eq. (4.14). In addition, the user must compute the initial conditions for an index-1 DAE initial value problem in which the state vector y cannot be partitioned into algebraic and differential variables [31].

4.2 An Index-1 Formulation of the Non-Centroidal DAE of Motion

Consider the non-centroidal formulation presented in Section A.6 of the Appendix. The resulting index-3 DAE of Eqs. (A.134) and (A.135), is

$$M(q, \beta, t)q'' + {\Phi_q}^\top \lambda = S^1(q, q', \beta, t) \tag{4.15}$$

$$\Phi(q, \beta, t) = 0 \tag{4.16}$$

where

$$S^1 = L + Q \tag{4.17}$$

is the sum of Coriolis force L and applied force Q. Given a consistent set of initial conditions; i.e., generalized coordinate vector q, generalized velocity vector $v \equiv q'$, generalized acceleration vector $a \equiv q''$, and Lagrange multiplier vector λ that satisfy at initial time t_0 the index-3 DAE of motion [27] of Eqs. (4.15) and (4.16), the velocity

constraint equation

$$\Phi_q q' + \Phi_t = 0 \tag{4.18}$$

obtained by differentiation of the position constraint equation of Eq. (4.16), and the acceleration constraint equation

$$\Phi_q q'' + (\Phi_q q')_q q' + 2\Phi_{q,t} q' + \Phi_{t,t} = 0 \tag{4.19}$$

obtained by differentiation of the velocity constraint equation of Eq. (4.18); the DAE of motion has a unique solution [27, 48] at all times.

An index-1 DAE that is equivalent to the DAE of motion of Eqs. (4.15) and (4.16) is obtained [2] by applying a stabilized index reduction method [24] to the DAE of motion and introducing differential variables χ and ψ,

$$F(y, y', t) = \begin{pmatrix} q' - v + {\Phi_q}^\top \chi' \\ M v' + {\Phi_q}^\top \psi' - S^1 \\ \Phi_q v + \Phi_t \\ \Phi(q, t) \end{pmatrix} = 0 \tag{4.20}$$

where

$$y = \begin{pmatrix} \chi \\ \psi \\ v \\ q \end{pmatrix} \tag{4.21}$$

The DAE of Eq. (4.20) has the advantage of satisfying both the position constraint equation of Eq. (4.16) and the velocity constraint equation of Eq. (4.18). As a result, after a numerical integration step applied to Eq. (4.20), projections of position and

velocity vectors onto position constraint and velocity constraint manifolds, respectively, are not necessary. A s-step variable step-size BDF method, $s \leq 6$, applied to the fully implicit index-1 DAE [2] of Eq. (4.20) converges with order s, as shown in Section 4.5. The Jacobian J of Eq. (4.14) that corresponds to the index-1 DAE of motion of Eq. (4.20) is

$$
\begin{aligned}
J &= F_y + \alpha_1 F_{y'} \\
&= \begin{pmatrix} 0 & 0 & -I & (\Phi_q{}^\top \chi')_q \\ 0 & 0 & -S_{1_v} & (Mv' + \Phi_q{}^\top \Psi' - S^1)_q \\ 0 & 0 & \Phi_q & (\Phi_q q')_q + \Phi_{q,t} \\ 0 & 0 & 0 & \Phi_q \end{pmatrix} \\
&+ \alpha_1 \begin{pmatrix} \Phi_q{}^\top & 0 & 0 & I \\ 0 & \Phi_q{}^\top & M & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \qquad (4.22) \\
&= \begin{pmatrix} \alpha_1 \Phi_q{}^\top & 0 & -I & (\Phi_q{}^\top \chi')_q + \alpha_1 I \\ 0 & \alpha_1 \Phi_q{}^\top & (\alpha_1 M - S_v^1) & (Mv' + \Phi_q{}^\top \Psi' - S^1)_q \\ 0 & 0 & \Phi_q & (\Phi_q q')_q + \Phi_{q,t} \\ 0 & 0 & 0 & \Phi_q \end{pmatrix} \qquad (4.23)
\end{aligned}
$$

Since the state vector y in the index-1 DAE equivalent formulation of Eq. (4.20) cannot be partitioned into separate differential and algebraic parts, consistent initial conditions cannot be directly evaluated by the IDA integrator. Therefore, calculation

of consistent initial conditions, together with evaluators of function F of Eq. (4.20) and Jacobian J of Eq. (4.22) must be provided separately. These evaluations require computation of the following derivatives:

1. Position constraint Jacobian Φ_q and its first and second derivatives, ${\Phi_q}'$ and ${\Phi_q}''$. Although the second derivative of the Jacobian is not used in implicit integration of the DAE of motion, its evaluation is required, as shown in Sections 4.3 and 4.4, for implicit integration of the DAE resulting from the direct differentiation and adjoint formulations.

2. Derivatives of terms of the form ${\Phi_q}^\top \gamma$ with respect to the generalized coordinate vector q, where γ is a constant vector.

3. Derivatives of terms of the form $M\gamma$ with respect to the generalized coordinate vector q, where γ is a constant vector.

4. Derivatives of vector $S^1 = L + Q$ with respect to the generalized coordinate vector q and its time derivative v.

4.2.1 Consistent Initial Conditions

Calculation of consistent initial conditions for the IVP of Eq. (4.20) starts with a given initial configuration of the multibody system; i.e., given initial positions and orientations of the bodies in the system, which define the initial generalized coordinate vector q_0. The process of obtaining a consistent initial configuration; i.e., an initial generalized coordinate vector q_0 that satisfies the position constraint of

Eq. (4.16), within a given tolerance, is known as assembly of the system. Numerical algorithms for multibody system assembly are presented by Haug [27]. Next, a set of initial velocities; i.e., the initial time derivative v_0 of the generalized coordinate vector must be given. In order to enforce the velocity constraint equation, an initial projection step [35] is performed using the generalized coordinate vector q_0 resulting from the assembly process. If the multibody system is assumed to start from a rest configuration, then the initial velocity is zero.

After consistent initial position q_0 and velocity v_0 vectors have been obtained, the equation of motion of Eq. (4.15) and the acceleration constraint of Eq. (4.19), at initial time t_0, define a linear system,

$$\begin{pmatrix} M_0 & {\Phi_{q0}}^\top \\ \Phi_{q0} & 0 \end{pmatrix} \begin{pmatrix} a_0 \\ \lambda_0 \end{pmatrix} = \begin{pmatrix} S_0^1 \\ -\left(\left(\Phi_q q'\right)_q v + 2\Phi_{q,t} v + \Phi_{t,t}\right)_0 \end{pmatrix} \tag{4.24}$$

where a_0, the initial acceleration vector, and λ_0, the initial vector of Lagrange multipliers, are unknowns and

$$M_0 = M(q_0, \beta, t_0)$$

$$\Phi_{q0} = \Phi_q(q_0, \beta, t_0)$$

$$S_0^1 = S^1(q_0, v_0, \beta, t_0)$$

$$\left(\left(\Phi_q q'\right)_q v + 2\Phi_{q,t} v + \Phi_{t,t}\right)_0 = \left(\left(\Phi_q q'\right)_q v + 2\Phi_{q,t} v + \Phi_{t,t}\right)(q_0, v_0, \beta, t_0)$$

Since the multibody system mass matrix is positive definite on the null-space of the constraint Jacobian [27], the coefficient matrix M^S of the linear system of Eq. (4.24),

known as the Schur-complement and defined by

$$M^S(q,\beta,t) \equiv \begin{pmatrix} M & {\Phi_q}^\top \\ \Phi_q & 0 \end{pmatrix} \tag{4.25}$$

is non-singular [52]. Therefore, initial accelerations and Lagrange multipliers are uniquely obtained by solving the linear system of Eq. (4.24). As a result, initial conditions y_0 and ${y'}_0$ for the index-1 DAE of Eq. (4.20) are

$$y_0 = \begin{pmatrix} \chi_0 \\ \psi_0 \\ v_0 \\ q_0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ v_0 \\ q_0 \end{pmatrix} \tag{4.26}$$

and

$${y'}_0 = \begin{pmatrix} {\chi'}_0 \\ {\psi'}_0 \\ a_0 \\ v_0 \end{pmatrix} = \begin{pmatrix} 0 \\ \lambda_0 \\ a_0 \\ v_0 \end{pmatrix} \tag{4.27}$$

4.2.2 Analytic Evaluation of the Position Constraint Jacobian and its First and Second Derivatives

For a multibody system consisting of n_b bodies, the generalized coordinate vector q is

$$q \equiv ({q_1}^\top \dots {q_{n_b}}^\top)^\top \tag{4.28}$$

The position and orientation of body i are defined by the seven-dimensional vector $q_i = \begin{pmatrix} {r_i}^\top & {p_i}^\top \end{pmatrix}^\top$, where r_i is the three-dimensional vector that defines the position of the origin of a body i non-centroidal body frame with respect to a Cartesian coordinate global coordinate frame and p_i is the four-dimensional normalized vector comprising the Euler parameters e_0, e_1, e_2, and e_3 that define the orientation of body i with respect to the global coordinate frame [27]. Matrix A_i [27] is the 3×3 orientation matrix that defines the orientation of body i in terms of Euler parameter vector p_i. Vector ${s'}_i^P$ is the three-dimensional vector that defines the position of a point P on body i with respect to body i's body fixed coordinate frame.

The first derivative of the constraint Jacobian, as shown in Eq. (A.23) of the Appendix, is

$$\Phi_q{}' = \left(\Phi_q q'\right)_q + \Phi_{q,t} \tag{4.29}$$

Using differential operator Θ introduced in Eq. (3.63) of Chapter 3,

$$\Theta(\Phi, \gamma) \equiv \left(\Phi_q \gamma\right)_q \tag{4.30}$$

where γ is a vector not depending on q, Eq. (4.29) is re-written as

$$\Phi_q{}' = \Theta(\Phi, q') + \Phi_{q,t} \tag{4.31}$$

The second derivative of the constraint Jacobian, as shown in Eq. (A.39) of the Appendix, is

$$\begin{aligned} \Phi_q{}'' &= \Phi_{q,tt} + \left(\Phi_{q,t} q'\right)_q + \left(\Phi_q q'\right)_{q,t} \\ &+ \left(\left(\Phi_q q'\right)_q q'\right)_q + \left(\Phi_q q''\right)_q \end{aligned} \tag{4.32}$$

Using differential operators Θ and Υ introduced in Eq. (3.76) of Chapter 3,

$$\Upsilon(\Phi, \gamma_1, \gamma_2) \equiv ((\Phi_q \gamma_1)_q \gamma_2)_q = ((\Theta(\Phi, \gamma_1))\gamma_2)_q \tag{4.33}$$

where $\gamma_1$and γ_2 are vectors that do not depend on q, Eq. (4.32) is re-written as

$$\begin{aligned} \Phi_q{}'' &= \Phi_{q,tt} + (\Phi_{q,t} q')_q + (\Phi_q q')_{q,t} \\ &+ \Upsilon(\Phi, q', q') + \Theta(\Phi, q'') \end{aligned} \tag{4.34}$$

The position constraint function Φ consists of (1) algebraic constraints introduced by joints between bodies and (2) Euler parameter normalization constraints,

$$\Phi = \begin{pmatrix} \bar{\Phi} \\ \\ \Phi^P \end{pmatrix} \tag{4.35}$$

Joint constraints are a combination of one or more of the following basic kinematic constraints [27]:

1. Spherical constraint $\Phi^S = 0$
2. Dot-1 constraint $\Phi^{d_1} = 0$
3. Dot-2 constraint $\Phi^{d_2} = 0$
4. Distance constraint $\Phi^D = 0$

Since none of the four basic kinematic constraint functions depend explictly on time, first and second derivative of their Jacobian are expressed only in terms of operators Θ and Υ, as follows:

$$\Phi_q^{K\prime} = \Theta(\Phi^K, q') \tag{4.36}$$

and

$$\Phi_q^{K''} = \Upsilon(\Phi^K, q', q') + \Theta(\Phi^K, q'') \tag{4.37}$$

where $K \in \{S, d_1, d_2, D\}$. Therefore, for a joint consisting of any combination of the four basic constraints, in addition to evaluating its Jacobian, calculating corresponding Θ and Υ operators is sufficient for computing the Jacobian first and second derivatives.

The spherical constraint between bodies i and j is defined [27] as

$$\Phi^S = r_i + A_i s'^P_i - (r_j + A_j s'^P_j) = 0 \tag{4.38}$$

The Jacobian of the spherical constraint; i.e., the derivative of Φ^S with respect to q_i and q_j is [51]

$$\Phi^S_{q_i,q_j} = \begin{pmatrix} I_3 & B(p_i, s'^P_i) & -I_3 & -B(p_j, s'^P_j) \end{pmatrix} \tag{4.39}$$

where three-dimensional vector s'^P_i is the position of a point fixed in body i, with respect to the body i coordinate frame, and s'^P_j is the position of a point fixed in body j, with respect to body j coordinate frame. Matrix $B(p, \gamma)$, where p is a normalized four-dimensional vector and γ a three-dimensional vector, is the derivative with respect to vector p of the product of the orientation matrix $A(p)$ and a vector γ and is defined in Ref. [51]. Operator Θ for the spherical constraint is

$$\Theta^S(\Phi^S, \gamma) = (\Phi^S_{q_i,q_j}\gamma)_{q_i,q_j} = \begin{pmatrix} 0 & B(\gamma^{i,p}, s'^P_i) & 0 & -B(\gamma^{j,p}, s'^P_j) \end{pmatrix} \tag{4.40}$$

where γ is a fourteen-dimensional vector partitioned into seven-dimensional compo-

nents γ^i and γ^j,

$$\gamma = \begin{pmatrix} \gamma^i \\ \\ \gamma^j \end{pmatrix} \tag{4.41}$$

and each seven-dimensional component k is partitioned into a three-dimensional component $\gamma^{k,r}$ and a four-dimensional component $\gamma^{k,p}$,

$$\gamma^k = \begin{pmatrix} \gamma^{k,r} \\ \\ \gamma^{k,p} \end{pmatrix}, k = i, j \tag{4.42}$$

Since vectors γ, ${s'}_i^P$, and ${s'}_j^P$ do not depend on the generalized coordinate vector q, it follows that $\Theta^S(\Phi^S, \gamma)$ is constant. As a result, operator Υ for the spherical constraint is

$$\Upsilon^S(\Phi^S, \gamma_1, \gamma_2) = \left(\Theta^S(\Phi^S, \gamma_1)\gamma_2\right)_{q_i, q_j} = 0 \tag{4.43}$$

In addition to operators Θ and Υ, let operator $\Delta\gamma$ be defined as

$$\begin{aligned} \Delta\gamma = \Phi^S_{q_i,q_j}\gamma &= \gamma^{i,r} + B(p_i, {s'}_i^p)\gamma^{i,p} - \gamma^{j,r} - B(p_j, {s'}_j^p)\gamma^{j,p} \\ &= \Phi^S_{q_i}\gamma^i - \Phi^S_{q_j}\gamma^j \end{aligned} \tag{4.44}$$

where γ is a fourteen-dimensional constant vector of the form of Eq. (4.41) and Φ^S is the constraint Jacobian of the spherical constraint. The derivative of operator $\Delta\gamma$ with respect to generalized coordinate components q_i and q_j is

$$\Delta\gamma_{q_i,q_j} = \begin{pmatrix} 0 & B(\gamma^{i,p}, {s'}_i^P) & 0 & -B(\gamma^{j,p}, {s'}_j^P) \end{pmatrix} \tag{4.45}$$

The dot-1 constraint between bodies i and j is defined [27] as

$$\Phi^{d_1} = ({A_i h'}_i)^\top {A_j h'}_j \tag{4.46}$$

where h'_i is a unit vector fixed with respect to the body i coordinate frame, and h'_j is a unit vector fixed with respect to the body j coordinate frame. The Jacobian of the dot-1 constraint is [51]

$$\Phi^{d_1}_{q_i,q_j} = \begin{pmatrix} 0 & (A_j h'_j)^\top B(p_i, h'_i) & 0 & (A_i h'_i)^\top B(p_j, h'_j) \end{pmatrix} \tag{4.47}$$

A fourteen-dimensional vector $\gamma = \begin{pmatrix} \gamma^{i\top} & \gamma^{j\top} \end{pmatrix}^\top$ is re-written, using Eq. (4.42), as

$$\gamma = \begin{pmatrix} (\gamma^{i,r})^\top & (\gamma^{i,p})^\top & (\gamma^{j,r})^\top & (\gamma^{j,p})^\top \end{pmatrix}^\top \tag{4.48}$$

Consider post-multiplication of the dot-1 Jacobian by vector γ,

$$\begin{aligned} \Phi^{d_1}_{q_i,q_j} \begin{pmatrix} \gamma^{i,r} \\ \gamma^{i,p} \\ \gamma^{j,r} \\ \gamma^{j,p} \end{pmatrix} &= (A_j h'_j)^\top B(p_i, h'_i)\gamma^{i,p} + (A_i h'_i)^\top B(p_j, h'_j)\gamma^{j,p} \\ &= \gamma^{i,p\top} B^\top(p_i, h'_i) A_j h'_j + \gamma^{j,p\top} B^\top(p_j, h'_j) A_i h'_i \end{aligned} \tag{4.49}$$

Since the product of the dot-1 Jacobian and vector γ does not depend on position vectors r_i and r_j, it follows that

$$(\Phi^{d_1}_{q_i,q_j}\gamma)_{r_i,r_j} = 0 \tag{4.50}$$

Differentiating the product of the dot-1 Jacobian and vector γ with respect to Euler parameter vectors p_i and p_j, the following matrices are obtained:

$$(\Phi^{d_1}_{q_i,q_j}\gamma)_{p_i} = (A_j h'_j)^\top B(\gamma^{i,p}, h'_i) + \gamma^{j,p\top} B^\top(p_j, h'_j) B(p_i, h'_i) \tag{4.51}$$

$$(\Phi^{d_1}_{q_i,q_j}\gamma)_{p_j} = \gamma^{i,p\top} B^\top(p_i, h'_i) B(p_j, h'_j) + (A_i h'_i)^\top B(\gamma^{j,p}, h'_j) \tag{4.52}$$

As a result of Eqs. (4.50), (4.51), and (4.52), operator Θ for the dot-1 constraint is

$$\Theta^{d_1}(\Phi^{d_1},\gamma)=\begin{pmatrix} 0 & (\Phi^{d_1}_{q_i,q_j}\gamma)_{p_i} & 0 & (\Phi^{d_1}_{q_i,q_j}\gamma)_{p_j}\end{pmatrix} \tag{4.53}$$

Next, consider post-multiplying matrix $\Theta^{d_1}(\Phi,\gamma)$ by a fourteen-dimensional constant vector ζ. Differentiating with respect to position vectors r_i and r_j and Euler parameter vectors p_i and p_j, the following matrices are obtained:

$$(\Theta^{d_1}(\Phi^{d_1},\gamma)\zeta)_{r_i,r_j}=0 \tag{4.54}$$

$$\begin{aligned}(\Theta^{d_1}(\Phi^{d_1},\gamma)\zeta)_{p_i} &= \gamma^{j,p^\top}B^\top(p_j,h'_j)B(\zeta^{i,p},h'_i)+\zeta^{j,p^\top}B^\top(p_j,h'_j)B(\gamma^{i,p},h'_i)\\ &+ \zeta^{j,p^\top}B^\top(\gamma^{j,p},h'_j)B(p_i,h'_i)\end{aligned} \tag{4.55}$$

$$\begin{aligned}(\Theta^{d_1}(\Phi^{d_1},\gamma)\zeta)_{p_j} &= \zeta^{i,p^\top}B^\top(\gamma^{i,p},h'_i)B(p_j,h'_j)+\zeta^{i,p^\top}B^\top(p_i,h'_i)B(\gamma^{j,p},h'_j)\\ &+ \gamma^{i,p^\top}B^\top(p_i,h'_i)B(\zeta^{j,p},h'_j)\end{aligned} \tag{4.56}$$

Therefore, operator Υ for the dot-1 constraint is

$$\Upsilon^{d_1}(\Phi^{d_1},\gamma,\zeta)=\begin{pmatrix} 0 & (\Theta^{d_1}(\Phi,\gamma)\zeta)_{p_i} & 0 & (\Theta^{d_1}(\Phi,\gamma)\zeta)_{p_j}\end{pmatrix} \tag{4.57}$$

The dot-2 constraint between bodies i and j is defined [27] as

$$\Phi^{d_2}=(A_ih'_i)^\top d_{ij}=0 \tag{4.58}$$

where h'_i is a three-dimensional unit vector fixed in the body i coordinate frame and the three-dimensional vector

$$d_{ij}=r_j+A_js'^P_j-r_i-A_is'^P_i=-\Phi^S \tag{4.59}$$

is the vector between a point fixed on body j, whose position with respect to its body coordinate frame is ${s'}_j^P$, and a point fixed on body i, whose position with respect to its body coordinate frame is ${s'}_i^P$. Since $d_{ij} = -\Phi^S$ the derivative of d_{ij} with respect to q_i and q_j is

$$(d_{ij})_{q_i,q_j} = -\Phi^S_{q_i,q_j} \tag{4.60}$$

The Jacobian of the dot-2 constraint is [51]

$$\Phi^{d_2}_{q_i,q_j} = -(A_i {h'}_i)^\top \Phi^S_{q_i,q_j} + {d_{ij}}^\top \begin{pmatrix} 0 & B(p_i, {h'}_i) & 0 & 0 \end{pmatrix} \tag{4.61}$$

Multiplying the Jacobian by the constant vector γ

$$\begin{aligned} (\Phi^{d_2}_{q_i,q_j}\gamma) &= -(A_i {h'}_i)^\top \Delta\gamma + {d_{ij}}^\top B(p_i, {h'}_i)\gamma^{i,p} \\ &= -(A_i {h'}_i)^\top \Delta\gamma + {\gamma^{i,p}}^\top B^\top(p_i, {h'}_i) d_{ij} \end{aligned} \tag{4.62}$$

and differentiating with respect to position and orientation vectors

$$(\Phi^{d_2}_{q_i,q_j}\gamma)_{r_i} = -{\gamma^{i,p}}^\top B^\top(p_i, {h'}_i) \tag{4.63}$$

$$(\Phi^{d_2}_{q_i,q_j}\gamma)_{r_j} = {\gamma^{i,p}}^\top B^\top(p_i, {h'}_i) \tag{4.64}$$

$$\begin{aligned} (\Phi^{d_2}_{q_i,q_j}\gamma)_{p_i} &= -\Delta\gamma^\top B(p_i, {h'}_i) - (A_i {h'}_i)^\top B(\gamma^{i,p}, {s'}_i^p) \\ &+ {d_{ij}}^\top B(\gamma^{i,p}, {h'}_i) - {\gamma^{i,p}}^\top B^\top(p_i, {h'}_i) B(p_i, {s'}_i^p) \end{aligned} \tag{4.65}$$

$$(\Phi^{d_2}_{q_i,q_j}\gamma)_{p_j} = (A_i {h'}_i)^\top B(\gamma^{j,p}, {s'}_j^p) + {\gamma^{i,p}}^\top B^\top(p_i, {h'}_i) B(p_j, {s'}_j^p) \tag{4.66}$$

The dot-2 Θ operator is

$$\Theta^{d_2}(\Phi^{d_2}, \gamma) = \begin{pmatrix} (\Phi^{d_2}_{q_i,q_j}\gamma)_{r_i} & (\Phi^{d_2}_{q_i,q_j}\gamma)_{p_i} & (\Phi^{d_2}_{q_i,q_j}\gamma)_{r_j} & (\Phi^{d_2}_{q_i,q_j}\gamma)_{p_j} \end{pmatrix} \tag{4.67}$$

Multiplying Θ^{d_2} by a constant fourteen-dimensional vector ζ and differentiating with respect to position and orientation vectors,

$$(\Theta^{d_2}(\Phi^{d_2},\gamma)\zeta)_{r_i} = -{\zeta^{i,p}}^{\top} B^{\top}(\gamma^{i,p}, {h'}_i) \tag{4.68}$$

$$(\Theta^{d_2}(\Phi^{d_2},\gamma)\zeta)_{r_j} = {\zeta^{i,p}}^{\top} B^{\top}(\gamma^{i,p}, {h'}_i) \tag{4.69}$$

$$\begin{aligned}
(\Theta^{d_2}(\Phi^{d_2},\gamma)\zeta)_{p_i} &= -{\zeta^{i,r}}^{\top} B(\gamma^{i,p}, {h'}_i) - \Delta{\gamma}^{\top} B(\zeta^{i,p}, {h'}_i) \\
&- {\zeta^{i,p}}^{\top} B^{\top}(p_i, {h'}_i) B(\gamma^{i,p}, {s'}_i^p) - {\zeta^{i,p}}^{\top} B^{\top}(\gamma^{i,p}, {s'}_i^p) B(p_i, {h'}_i) \\
&- {\zeta^{i,p}}^{\top} B^{\top}(\gamma^{i,p}, {h'}_i) B(p_i, {s'}_i^p) - {\gamma^{i,p}}^{\top} B^{\top}(p_i, {h'}_i) B(\zeta^{i,p}, {s'}_i^p) \\
&- {\zeta^{i,p}}^{\top} B^{\top}(p_i, {s'}_i^p) B(\gamma^{i,p}, {h'}_i) + {\zeta^{j,r}}^{\top} B(\gamma^{i,p}, {h'}_i) \\
&+ {\zeta^{j,p}}^{\top} B^{\top}(\gamma^{j,p}, {s'}_j^p) B(p_i, {h'}_i) + {\zeta^{j,p}}^{\top} B^{\top}(p_j, {s'}_j^p) B(\gamma^{i,p} {h'}_i)
\end{aligned} \tag{4.70}$$

$$\begin{aligned}
(\Theta^{d_2}(\Phi^{d_2},\gamma)\zeta)_{p_j} &= {\zeta^{i,p}}^{\top} B^{\top}(\gamma^{i,p}, {h'}_i) B(p_j, {s'}_j^p) + {\gamma^{i,p}}^{\top} B^{\top}(p_i, {h'}_i) B(\zeta^{j,p}, {s'}_j^p) \\
&+ {\zeta^{i,p}}^{\top} B^{\top}(p_i, {h'}_i) B(\gamma^{j,p}, {s'}_j^p)
\end{aligned} \tag{4.71}$$

the dot-2 Υ operator is

$$\Upsilon^{d_2}(\Phi^{d_2},\gamma,\zeta) = \begin{pmatrix} (\Theta^{d_2}(\Phi^{d_2},\gamma)\zeta)_{r_i}{}^{\top} \\ (\Theta^{d_2}(\Phi^{d_2},\gamma)\zeta)_{p_i}{}^{\top} \\ (\Theta^{d_2}(\Phi^{d_2},\gamma)\zeta)_{r_j}{}^{\top} \\ (\Theta^{d_2}(\Phi^{d_2},\gamma)\zeta)_{p_j}{}^{\top} \end{pmatrix}^{\top} \tag{4.72}$$

The distance constraint between bodies i and j is defined [27] as

$$\Phi^D = {d_{ij}}^{\top} d_{ij} - c^2 = 0 \tag{4.73}$$

where d_{ij} is the distance vector introduced by Eq. (4.59) and c is a scalar. The Jacobian of the distance constraint is [51]

$$\Phi^D_{q_i,q_j} = -2{d_{ij}}^\top \Phi^S_{q_i,q_j} = -2{d_{ij}}^\top \begin{pmatrix} I_3 & B(p_i, {s'}^P_i) & -I_3 & -B(p_j, {s'}^P_j) \end{pmatrix} \tag{4.74}$$

The distance Θ operator is

$$\begin{aligned} \Theta^D(\Phi^D, \gamma) &= -2\gamma^\top ({\Phi^S_{q_i,q_j}}^\top d_{ij})_{q_i,q_j} \\ &= 2\gamma^\top {\Phi^S_{q_i,q_j}}^\top \Phi^S_{q_i,q_j} - 2{d_{ij}}^\top (\Delta\gamma_{q_i,q_j}) \end{aligned} \tag{4.75}$$

Multiplying Θ^D by a constant fourteen-dimensional vector ζ,

$$\Theta^D(\Phi^D, \gamma)\zeta = 2\zeta^\top {\Phi^S_{q_i,q_j}}^\top \Delta\gamma - 2\zeta^\top (\Delta\gamma_{q_i,q_j})^\top d_{ij} \tag{4.76}$$

and differentiating with respect to generalized coordinate components q_i and q_j, the distance constraint Υ operator is

$$\Upsilon^D(\Phi^D, \gamma, \zeta) = (\Theta^D(\Phi^D, \gamma)\zeta)_{q_i,q_j} = 2\zeta^\top {\Phi^S_{q_i,q_j}}^\top \Delta\gamma_{q_i,q_j} + 2\Delta\gamma^\top \Delta\zeta_{q_i,q_j} + 2\zeta^\top (\Delta\gamma_{q_i,q_j})^\top \Phi^S_{q_i,q_j} \tag{4.77}$$

The Euler parameter normalization constraint for body i is defined [27] as

$$\Phi^P_i = {p_i}^\top p_i - 1 \tag{4.78}$$

The Jacobian of the Euler parameter normalization constraint is [27]

$${\Phi^P_i}_{q_i,q_j} = \begin{pmatrix} 0 & 2{p_i}^\top & 0 & 0 \end{pmatrix} \tag{4.79}$$

Multiplying the Jacobian by the constant vector γ and differentiating with respect to generalized coordinate components q_i and q_j, operator Θ for the Euler parameter

normalization constraint is obtained

$$\Theta_i^P(\Phi_i^P, \gamma) = (\Phi_{i\ q_i,q_j}^P \gamma)_{q_i,q_j} = (2{p_i}^\top \gamma^{i,p})_{q_i,q_j} = \begin{pmatrix} 0 & 2\gamma^{i,p\top} & 0 & 0 \end{pmatrix} \tag{4.80}$$

which is a constant matrix. As a result, operator Υ for the Euler parameters normalization constraint is

$$\Upsilon_i^P(\Phi_i^P, \gamma, \zeta) = (\Theta_i^P(\Phi_i^P, \gamma)\zeta)_{q_i,q_j} = 0 \tag{4.81}$$

4.2.3 Derivatives of the Product of the Transpose of Position Constraint Jacobian with a Vector

Consider the product

$${\Phi_q}^\top \zeta$$

where ζ does not depend on q, and let u_i be the unit vector with 1 in the i-th position and all the other elements zero. The product

$$\xi_i = {u_i}^\top ({\Phi_q}^\top \zeta)_q$$

is the i-th row of the matrix $({\Phi_q}^\top \zeta)_q$. Since u_i is a constant vector, ξ_i is re-written as

$$\xi_i = ({u_i}^\top {\Phi_q}^\top \zeta)_q$$

Since ${u_i}^\top {\Phi_q}^\top \zeta$ is a scalar,

$${u_i}^\top {\Phi_q}^\top \zeta = ({u_i}^\top {\Phi_q}^\top \zeta)^\top = \zeta^\top \Phi_q u_i$$

Since ζ does not depend on q, the i-th row of the matrix $({\Phi_q}^\top \zeta)_q$ is

$$\xi_i = (\zeta^\top \Phi_q u_i)_q = \zeta^\top (\Phi_q u_i)_q = \zeta^\top \Theta(\Phi, u_i) \tag{4.82}$$

Therefore, the matrix $(\Phi_q{}^\top \zeta)_q$ is

$$(\Phi_q{}^\top \zeta)_q = \begin{pmatrix} \zeta^\top \Theta(\Phi, u_1) \\ \vdots \\ \zeta^\top \Theta(\Phi, u_n) \end{pmatrix} \tag{4.83}$$

where each row is computed independently.

4.2.4 Derivatives of the Product of the Mass Matrix with a Vector

For a multibody system comprised of n_b bodies, in which orientation of bodies is defined using Euler parameters and body-fixed coordinate frames are non-centroidal, the mass matrix is a singular block-diagonal symmetric matrix, as shown in Eq. (A.131) of the Appendix,

$$M = diag(M_i, i = 1, 2, \ldots, n_b)$$

where the 7×7 matrix block M_i is given by Eq. (A.126) of the Appendix,

$$M_i(q_i, \beta) = \begin{pmatrix} m_i I_3 & -2m_i A_i \tilde{s'}_i^C G_i \\ 2m_i G_i{}^\top \tilde{s'}_i^C A_i{}^\top & 4G_i{}^\top J'_i G_i \end{pmatrix} \tag{4.84}$$

where m_i is the mass of body i, I_3 is the 3×3 identity matrix; $A_i(p_i)$ is the body i orientation matrix [27], $G_i = G(p_i)$ is an Euler parameter dependent matrix [27], $\tilde{s'}_i^C$ is the matrix obtained by applying the $\sim$ operator [27] to the i-th body center of mass position vector s'^C_i with respect to the body-fixed coordinate frame, and J'_i is the inertia tensor of body i with respect to its body-fixed coordinate frame.

The derivative of the product of matrix M_i and a constant vector

$$\gamma^i = \begin{pmatrix} \gamma^{i,r\top} & \gamma^{i,p\top} \end{pmatrix}^\top$$

with respect to q_i, defined by Eq. (A.146) of the Appendix, is

$$(M_i\gamma^i)_{q_i} = \begin{pmatrix} 0 & M_{1,i} \\ 0 & M_{2,i} \end{pmatrix} \tag{4.85}$$

where

$$M_{1,i} = -2m_iB(p_i, \tilde{s'}_i^C G_i\gamma^{i,p}) + 2m_iA_i\tilde{s'}_i^C G(\gamma^{i,p})$$

$$M_{2,i} = 2m_i(\tilde{s'}_i^C A_i{}^\top\gamma^{i,r})^- + 2m_iG_i{}^\top\tilde{s'}_i^C C(p_i, \gamma^{i,r}) + 4(J'_iG_i\gamma^{i,p})^- - 4G_i{}^\top J'_iG(\gamma^{i,p})$$

Operator $()^-$ is defined in Ref. [43] and matrix $C(p_i, \xi)$, depending on Euler parameter vector p_i and vector ξ, is defined in Ref. [50]. As a result, the matrix $(M\gamma)_q$, where $\gamma = \begin{pmatrix} \gamma_1{}^\top & \dots & \gamma_{n_b}{}^\top \end{pmatrix}^\top$, is

$$(M\gamma)_q = diag((M_i\gamma^i)_{q_i}, i = 1, 2, \dots, n_b) \tag{4.86}$$

and blocks $(M_i\gamma^i)_{q_i}$ are computed independently.

4.2.5 Derivatives of Applied Force and Coriolis Related Terms

The Coriolis force relative to a non-centroidal reference frame, defined in Eq. (A.132) of the Appendix, is

$$L = \begin{pmatrix} L_1{}^\top & \dots & L_{n_b}{}^\top \end{pmatrix}^\top$$

where component L_i, defined in Eq. (A.128) of the Appendix, is

$$L_i = \begin{pmatrix} 4m_i E_i {G_i'}^{\top} G_i' {G_i}^{\top} {s_i'}^C \\ -8{G_i'}^{\top} {J'}_i G_i p_i' \end{pmatrix} \tag{4.87}$$

where matrix E_i, which depends on the body i Euler parameter vector p_i, is defined in Ref. [27]. The vector function L_i depends only on the body i Euler parameter vector and its derivative. Therefore, the derivative of L_i with respect to $q_i = \begin{pmatrix} {r_i}^{\top} & {p_i}^{\top} \end{pmatrix}^{\top}$, defined by Eq. (A.153) of the Appendix, is

$$L_{iq_i} = \begin{pmatrix} 0 & -4m_i E({G_i'}^{\top} G_i' {G_i}^{\top} {s_i'}^C) + 4m_i E_i {G_i'}^{\top} G_i' ({s_i'}^C)^{-} \\ 0 & 8{G_i'}^{\top} {J'}_i G_i' \end{pmatrix} \tag{4.88}$$

and the derivative of L_i with respect to ${q_i}'$ is

$$L_{iq_i'} = \begin{pmatrix} 0 & 4m_i E_i (G_i' {G_i}^{\top} {s_i'}^C)^{-} - 4m_i E_i {G_i'}^{\top} G({G_i}^{\top} {s_i'}^C) \\ 0 & -8({J'}_i G_i p_i')^{-} - 8{G_i'}^{\top} {J'}_i G_i \end{pmatrix} \tag{4.89}$$

As a result, matrices L_q and $L_{q'}$ are

$$L_q = diag(L_{iq_i}, i = 1, 2, \ldots, n_b)$$

$$L_{q'} = diag(L_{iq_i'} i = 1, 2, \ldots, n_b)$$

where matrix blocks L_{iq_i} and $L_{iq_i'}$ are computed independently.

Generalized force Q_i acting on body i is of the form [50]

$$Q_i = \begin{pmatrix} F_i^A \\ 2{G_i}^{\top} {n'}_i^A \end{pmatrix} + Q_i^{TSDA} + Q_i^{RSDA} \tag{4.90}$$

where applied force F_i^A is given and torque ${n'}_i^A$ is defined in Eq. (A.110) of the Appendix as a function of F_i^A. The term Q_i^{TSDA} represents the force produced by a

translational spring-damper-actuator (TSDA) that connects bodies i and j and has the form [27]

$$Q_i^{TSDA} = \left(\frac{k(l - l_0) + cl' + f(l, l')}{l} \right) \begin{pmatrix} d_{ij} \\ \tilde{s'^P_i} A_j{}^\top d_{ij} \end{pmatrix} \tag{4.91}$$

where d_{ij}, defined by Eq. (4.59), is the vector between points s'^P_i and s'^P_j on bodies i and j, respectively; $l = \|d_{ij}\|$ is the distance between the two points; l_0 is the given distance at the spring rest position; k is the given spring stiffness; c is the given damping constant; and $f(l, l')$ is the known actuator force function, depending on l and its derivative l'. The term Q_i^{RSDA} represents the force produced by a rotational spring-damper-actuator (RSDA) that connects bodies i and j and has the form [27]

$$Q_i^{RSDA} = (k_\theta(\theta + 2n_{rev}\pi) + c_\theta\theta' + n(\theta + 2n_{rev}\pi, \theta')) \begin{pmatrix} 0 \\ h'_i \end{pmatrix} \tag{4.92}$$

where h'_i is a unit vector fixed with respect to the body i coordinate frame, θ is the angle between the rotational spring rest position and current position $0 \leq \theta \leq 2\pi$, n_{rev} is the integer number of revolutions of the rotational spring relative to its rest position, k_θ is the given rotational spring stiffness constant, c_θ is the rotational damper constant, and $n(\theta, \theta')$ is the rotational actuator known torque, depending on θ and its derivative θ'.

The applied force Q_i depends on generalized coordinate components q_i and q_j

and their derivatives $q_i{}'$ and $q_j{}'$. Its derivative with respect to $q_{ij} = \begin{pmatrix} q_i{}^\top & q_j{}^\top \end{pmatrix}^\top$ is

$$\begin{aligned} Q_{i_{q_{ij}}} &= \begin{pmatrix} (F_i^A)_{q_{ij}} \\ 2G_i{}^\top n'^A_i{}_{q_{ij}} \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 2(n'^A_i)^- & 0 & 0 \end{pmatrix} \\ &+ Q_i^{TSDA}{}_{q_{ij}} + Q_i^{RSDA}{}_{q_{ij}} \end{aligned} \tag{4.93}$$

and the derivative of the applied force with respect to $q'_{ij} = \begin{pmatrix} q'_i{}^\top & q'_j{}^\top \end{pmatrix}^\top$ is

$$Q_{iq'_{ij}} = \begin{pmatrix} (F_i^A)_{q'_{ij}} \\ 2G_i{}^\top n'^A_i{}_{q_{ij}} \end{pmatrix} + Q_i^{TSDA}{}_{q'_{ij}} + Q_i^{RSDA}{}_{q'_{ij}} \tag{4.94}$$

where derivatives $Q_i^{TSDA}{}_{q_{ij}}$, $Q_i^{TSDA}{}_{q'_{ij}}$, $Q_i^{RSDA}{}_{q_{ij}}$, and $Q_i^{RSDA}{}_{q'_{ij}}$ are defined by Eqs. (A.172) through (A.202) of the Appendix. As a result, derivatives Q_q and $Q_{q'}$ of the applied force

$$Q = \begin{pmatrix} Q_1{}^\top & \dots & Q_{n_b}{}^\top \end{pmatrix}^\top$$

are obtained by independently computing blocks $Q_{iq_{ij}}$ and $Q_{iq'_{ij}}$ and assembling them into matrices Q_q and $Q_{q'}$.

4.3 An Index-1 Formulation of the Direct Differentiation DAE

Differentiating the equation of motion of Eq. (4.15) with respect to design parameter β_l, $l \in \{1, 2, \dots, n_\beta\}$, where β is the vector of design parameters,

$$\beta = \begin{pmatrix} \beta_1{}^\top & \dots & \beta_{n_\beta}{}^\top \end{pmatrix}$$

the direct differentiation index-3 DAE is obtained as [26]

$$\begin{aligned} M{q_{\beta_l}}'' + {\Phi_q}^\top \lambda_{\beta_l} &= \big(L_q + Q_q - (Mq'')_q - ({\Phi_q}^\top \lambda)_q\big) q_{\beta_l} + (L_{q'} + Q_{q'}) {q_{\beta_l}}' \\ &- (M\hat{q}'')_{\beta_l} - ({\Phi_q}^\top \hat{\lambda})_{\beta_l} + L_{\beta_l} + Q_{\beta_l} \qquad (4.95) \\ \Phi_q q_{\beta_l} + \Phi_{\beta_l} &= 0 \qquad (4.96) \end{aligned}$$

where Eq. (4.95) represents the differential sensitivity equation and Eq. (4.96) represents the position sensitivity constraint equation. Terms of the form $(A(\eta)\hat{\xi})_\eta$ denote that variable ξ is held fixed during differentiation with respect to variable η. Differentiating the position sensitivity constraint equation (4.96), the velocity sensitivity constraint equation is obtained as

$$\Phi_q {q_{\beta_l}}' + (\Phi_q)' q_{\beta_l} + (\Phi_{\beta_l})' = 0 \qquad (4.97)$$

and differentiating the velocity sensitivity constraint equation of Eq. (4.97), the acceleration sensitivity constraint equation is obtained as

$$\Phi_q {q_{\beta_l}}'' + 2(\Phi_q)' {q_{\beta_l}}' + (\Phi_q)'' q_{\beta_l} + (\Phi_{\beta_l})'' = 0 \qquad (4.98)$$

Defining variables

$$x^q(\beta, t) \equiv q_{\beta_l}(\beta, t) \qquad (4.99)$$

$$x^v(\beta, t) \equiv {q_{\beta_l}}'(\beta, t) \qquad (4.100)$$

$$z(\beta, t) \equiv \lambda_{\beta_l}(\beta, t) \qquad (4.101)$$

the direct differentiation DAE of Eqs. (4.95) and (4.96) is re-written as

$$\begin{aligned} M{x^v}' + {\Phi_q}^\top z &= s_d + Q^{x_q} x^q + Q^{x_v} x^v \qquad (4.102) \\ \Phi_q x^q &= r_d \qquad (4.103) \end{aligned}$$

where subscript d stands for direct differentiation method and functions s_d, r_d, Q^{x_q}, and Q^{x_v} are

$$s_d = S^1_{\beta_l} - (M\hat{q}''_{\beta_l})_{\beta_l} - ({\Phi_q}^\top \hat{\lambda})_{\beta_l} \tag{4.104}$$

$$r_d = -\Phi_{\beta_l} \tag{4.105}$$

$$Q^{x_q} = S^1_q - (Mq'')_q - ({\Phi_q}^\top \lambda)_q \tag{4.106}$$

$$Q^{x_v} = S^1_v \tag{4.107}$$

The velocity sensitivity constraint equation of Eq. (4.97) is re-written as

$$\Phi_q x^v + (\Phi_q)' x^q = {r_d}' \tag{4.108}$$

where

$${r_d}' = -(\Phi_{\beta_l})' = -(\Phi_q \hat{q}')_{\beta_l} - \Phi_{t,\beta_l} \tag{4.109}$$

and the acceleration sensitivity constraint equation of Eq. (4.98) is re-written as

$$\Phi_q x^{v\prime} + 2(\Phi_q)' x^v + (\Phi_q)'' x^q = {r_d}'' \tag{4.110}$$

where

$$\begin{aligned} {r_d}'' &= -(\Phi_{\beta_l})'' \\ &= -((\Phi_q \hat{q}')_q \hat{q}')_{\beta_l} - (\Phi_q \hat{q}'')_{\beta_l} - 2(\Phi_{q,t}\hat{q}')_{\beta_l} - \Phi_{tt,\beta_l} \end{aligned} \tag{4.111}$$

Defining the vector

$$Q^x = Q^{x_v} x^v + Q^{x_q} x^q \tag{4.112}$$

the equivalent index-1 DAE formulation [2], obtained by applying a stabilized index reduction [24] to the DAE of Eqs. (4.102) and (4.103), augmenting the resulting DAE

with the constraint equation of Eq. (4.108), and introducing variables χ_d and ψ_d is

$$F_d(y, y', t) = \begin{pmatrix} x^{q\prime} - x^v + {\Phi_q}^\top {\chi_d}' \\ M x^{v\prime} + {\Phi_q}^\top {\psi_d}' - Q^x - s_d \\ \Phi_q x^v + (\Phi_q)' x^q - {r_d}' \\ \Phi_q x^q - r_d \end{pmatrix} = 0 \tag{4.113}$$

where $y = \begin{pmatrix} {\chi_d}^\top & {\psi_d}^\top & x^{v\top} & x^{q\top} \end{pmatrix}^\top$. A variable step-size BDF integration method of up to order six converges for the index-1 DAE of Eq. (4.113), as is shown in Section 4.5. The Jacobian of the non-linear equation that results from applying a BDF integration method to the index-1 DAE of Eq. (4.113) is

$$J_d = \begin{pmatrix} \alpha_1 {\Phi_q}^\top & 0 & -I & \alpha_1 I \\ 0 & \alpha_1 {\Phi_q}^\top & \alpha_1 M - Q^{x_v} & -Q^{x_q} \\ 0 & 0 & \Phi_q & (\Phi_q)' \\ 0 & 0 & 0 & \Phi_q \end{pmatrix} \tag{4.114}$$

Evaluations of function F_d, Jacobian J_d, functions ${r_d}'$ and ${r_d}''$, and consistent initial conditions require computation of the following derivatives:

1. The position constraint Jacobian Φ_q and derivatives of the form $\left({\Phi_q}^\top \zeta\right)_q$, $\left(M\gamma\right)_q$, S_q^1, and S_v^1. The analytic evaluation of these derivatives is shown in Section 4.2.

2. Partial derivatives of kinematic terms with respect to design parameters β, Φ_β, $\left({\Phi_q}^\top \zeta\right)_\beta$, $\left(M\gamma\right)_\beta$, S_β^1, $\left(\Phi_q \gamma\right)_\beta$, and $\left(\left(\Phi_q \gamma\right)_q \zeta\right)_\beta$.

4.3.1 Consistent Initial Conditions

Consistent initial conditions for the direct differentiation DAE of Eqs. (4.95) and (4.96) are computed assuming one of the following hypothesis [50]:

1. The initial configuration of the multibody system depends explicitly upon the design parameters [26]. As a result, initial conditions for the sensitivity equations are obtained by direct differentiation of the initial conditions of equations of motion [26].

2. The generalized coordinate vector q is partitioned into subvectors q^u and q^v, $q = [q^{u\top} q^{v\top}]^\top$, where q^v has dimension equal to the number d of degrees of freedom of the system. Also, the corresponding subvector $q^{v\prime}$ of the generalized velocities is given. As a result, the sensitivities q^v_β and q^u_β, and their derivatives $q^{v\prime}_\beta$ and $q^{u\prime}_\beta$ are [50]

$$\begin{aligned} q^v_\beta &= 0 \\ q^u_\beta &= -(\Phi_u)^{-1}\Phi_\beta \\ q^{v\prime}_\beta &= 0 \\ q^{u\prime}_\beta &= -(\Phi_u)^{-1}\big(\Phi_{t,\beta} + (\Phi_q \hat{q}')_\beta + (\Phi_q \hat{q}' + \Phi_t)_q q_\beta\big) \end{aligned}$$

   where the constraint Jacobian Φ_q is partitioned into a nonsingular square block Φ_u and block Φ_v,

$$\Phi_q = \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix}$$

3. The initial configuration is a static equilibrium configuration.

   Therefore, generalized velocities q' and accelerations q'' are zero, and the initial

sensitivities are obtained [50] by solving the linear system

$$\begin{pmatrix} (\Phi_q{}^\top \hat{\lambda})_q - S_q^1 & \Phi_q{}^\top \\ \Phi_q & 0 \end{pmatrix} \begin{pmatrix} q_\beta \\ \lambda_\beta \end{pmatrix} = \begin{pmatrix} S_\beta^1 - (\Phi_q{}^\top \hat{\lambda})_\beta \\ -\Phi_\beta \end{pmatrix}$$

$$q_\beta{}' = 0$$

After consistent initial sensitivities for position q_{β_0} and velocity $q_\beta{}'_0$ have been obtained, the differential sensitivity equation of Eq. (4.96), together with the acceleration sensitivity constraint of Eq. (4.97), define the linear system

$$M_0^S w_0^d = b_0^d \tag{4.115}$$

where M_0^S is the Schur complement matrix, defined by Eq. (4.25), at initial time t_0, w_0^d is the vector of unknowns,

$$w_0^d = \begin{pmatrix} q_{\beta_l}{}''(t_0)^\top & \lambda_{\beta_l}(t_0)^\top \end{pmatrix}^\top$$

and

$$b_0^d = \begin{pmatrix} \big((L_q + Q_q - (Mq'')_q - (\Phi_q{}^\top \lambda)_q) q_{\beta_l} + (L_{q'} + Q_{q'}) q_{\beta_l}{}'\big)(t_0) \\ \big(- 2(\Phi_q)' q_{\beta_l}{}' - (\Phi_q)'' q_{\beta_l} - (\Phi_{\beta_l})''\big)(t_0) \end{pmatrix}$$

As a result, initial conditions y_0 and y'_0 for the index-1 DAE of Eq. (4.113) are

$$y_0 = \begin{pmatrix} \chi_{d0} \\ \psi_{d0} \\ q_\beta{}'_0 \\ q_{\beta_0} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ q_\beta{}'_0 \\ q_{\beta_0} \end{pmatrix} \tag{4.116}$$

and

$$y'_0 = \begin{pmatrix} {\chi_d}'_0 \\ {\psi_d}'_0 \\ {q_{\beta_l}}''(t_0) \\ {q_{\beta_l}}'(t_0) \end{pmatrix} = \begin{pmatrix} 0 \\ \lambda_{\beta_l}(t_0) \\ {q_{\beta_l}}''(t_0) \\ {q_{\beta_l}}'(t_0) \end{pmatrix} \tag{4.117}$$

4.3.2 Evaluation of Partial Derivatives of Kinematic Terms with respect to Design Parameters

Partial derivatives of kinematic terms with respect to design parameters β are obtained by using the chain rule of differentiation. For a kinematic term $X(q, q', q'', \lambda, u(\beta))$ that depends on the generalized coordinate vector q, its first and second derivatives q' and q'', Lagrange multipliers λ, and a set of model parameters $u(\beta)$ that are functions of design parameters β, the partial derivative

$$\frac{\partial X}{\partial \beta} = \frac{\partial X}{\partial u}\frac{\partial u}{\partial \beta}$$

has the following constituent parts:

1. The partial derivative of X with respect to model parameter u can be analytically evaluated, since, for a given topology, the multibody system depends explicitly on model parameters; e.g., positions of points in bodies with respect to corresponding body-fixed coordinate frame, inertia properties of a body, and body fixed unit vectors.

2. The partial derivative of the vector of model parameter u with respect to design

parameter β, which must be given, or evaluated through finite differences or the complex-step method.

In this thesis, only evaluation of terms in the first part is provided, while all the necessary terms in the second part are assumed to be explicitly given. Next, evaluation of derivatives with respect to model parameters of the form Φ_u, $(\Phi_q\gamma)_u$, $(\Phi_q{}^\top\zeta)_u$, and $((\Phi_q\gamma)_q\zeta)_u$, where γ and ζ are constant known vectors, is presented for each of the four basic constraints; spherical, dot-1, dot-2, and distance. Let constant fourteen-dimensional vectors γ, γ_1, and γ_2 have the form of Eq. (4.48), γ_3 be a three-dimensional constant vector, and α be a constant scalar.

The spherical constraint function Φ^S of Eq. (4.38) depends on model parameters ${s'}_i^P$ and ${s'}_j^P$. Derivatives ${\Phi^S}_u$, $({\Phi^S}_q\gamma)_u$, $({\Phi^S}_q{}^\top\zeta)_u$, and $(({\Phi^S}_q\gamma)_q\zeta)_u$, where γ and ζ are constant known vectors and $u \in \{{s'}_i^P, {s'}_j^P\}$, are [50]

$$\Phi^S_{{s'}_i^P} = A_i \tag{4.118}$$

$$\Phi^S_{{s'}_j^P} = -A_j \tag{4.119}$$

$$(\Phi^S_{q_i,q_j}\gamma)_{{s'}_i^P} = N(p_i, \gamma^{i,p}) \tag{4.120}$$

$$(\Phi^S_{q_i,q_j}\gamma)_{{s'}_j^P} = -N(p_j, \gamma^{j,p}) \tag{4.121}$$

$$(\Phi^S_{q_i,q_j}{}^\top\gamma_3)_{{s'}_i^P} = \begin{pmatrix} 0 \\ C^\top(p_i, \gamma_3) \\ 0 \\ 0 \end{pmatrix} \tag{4.122}$$

$$({\Phi^S_{q_i,q_j}}^\top \gamma_3)_{{s'}^P_j} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -C^\top(p_j, \gamma_3) \end{pmatrix} \tag{4.123}$$

$$\begin{aligned} \left((\Phi^S_{q_i,q_j}\gamma_1)_{q_i,q_j}\gamma_2\right)_{{s'}^P_i} &= \left(B(\gamma_1^{i,p}, {s'}^P_i)\gamma_2^{i,p} - B(\gamma_1^{j,p}, {s'}^P_j)\gamma_2^{j,p}\right)_{{s'}^P_i} \\ &= N(\gamma_1^{i,p}, \gamma_2^{i,p}) \end{aligned} \tag{4.124}$$

$$\begin{aligned} \left((\Phi^S_{q_i,q_j}\gamma_1)_{q_i,q_j}\gamma_2\right)_{{s'}^P_j} &= \left(B(\gamma_1^{i,p}, {s'}^P_i)\gamma_2^{i,p} - B(\gamma_1^{j,p}, {s'}^P_j)\gamma_2^{j,p}\right)_{{s'}^P_j} \\ &= -N(\gamma_1^{j,p}, \gamma_2^{j,p}) \end{aligned} \tag{4.125}$$

Derivatives

$$N(p_i, \gamma^{j,p}) \equiv \frac{\partial(B(p_i, {a'}_i)\gamma^{j,p})}{\partial {a'}_i} \tag{4.126}$$

and

$$C^\top(p_i, \gamma_3) \equiv \frac{\partial(B^\top(p_i, {a'}_i)\gamma_3)}{\partial {a'}_i} \tag{4.127}$$

are defined by Eqs. (A.98) and (A.101), respectively, of the Appendix.

The dot-1 constraint function Φ^{d_1} of Eq. (4.46) depends on model parameters ${h'}_i$ and ${h'}_j$. Derivatives ${\Phi^{d_1}}_u$, $({\Phi^{d_1}}_q\gamma)_u$, $({\Phi^{d_1}}_q{}^\top\zeta)_u$, and $(({\Phi^{d_1}}_q\gamma)_q\zeta)_u$, where γ and ζ are constant known vectors and $u \in \{{h'}_i, {h'}_j\}$, are [50]

$$\Phi^{d_1}_{{h'}_i} = {h'}_j{}^\top {A_j}^\top A_i \tag{4.128}$$

$$\Phi^{d_1}_{{h'}_j} = {h'}_i{}^\top {A_i}^\top A_j \tag{4.129}$$

$$(\Phi^{d_1}_q\gamma)_{{h'}_i} = (A_j {h'}_j)^\top N(p_i, \gamma^{p_i}) + \gamma^{p_j\top} B^\top(p_j, {h'}_j) A_i \tag{4.130}$$

$$(\Phi_q^{d_1}\gamma)_{{h'}_j} = (A_i {h'}_i)^\top N(p_j, \gamma^{p_j}) + {\gamma^{p_i}}^\top B^\top(p_i, {h'}_i) A_j \tag{4.131}$$

$$({\Phi_q^{d_1}}^\top \alpha)_{{h'}_i} = \alpha \begin{pmatrix} 0 \\ C^\top(p_i, A_j {h'}_j) \\ 0 \\ B^\top(p_j, {h'}_j) A_i \end{pmatrix} \tag{4.132}$$

$$({\Phi_q^{d_1}}^\top \alpha)_{{h'}_j} = \alpha \begin{pmatrix} 0 \\ B^\top(p_i, {h'}_i) A_j \\ 0 \\ C^\top(p_j, A_i {h'}_i) \end{pmatrix} \tag{4.133}$$

$$\begin{aligned} \left(\left({\Phi^{d_1}}_{q_i,q_j}\gamma_1\right)_{q_i,q_j}\gamma_2\right)_{{h'}_i} &= \Big((A_j {h'}_j)^\top B(\gamma_1^{i,p}, {h'}_i)\gamma_2^{i,p} + {\gamma_1^{j,p}}^\top B^\top(p_j, {h'}_j) B(p_i, {h'}_i)\gamma_2^{i,p} \\ &+ {\gamma_1^{i,p}}^\top B^\top(p_i, {h'}_i) B(p_j, {h'}_j)\gamma_2^{j,p} + (A_i {h'}_i)^\top B(\gamma_1^{j,p}, {h'}_j)\gamma_2^{j,p}\Big)_{{h'}_i} \\ &= (A_j {h'}_j)^\top N(\gamma_1^{i,p}, \gamma_2^{i,p}) + {\gamma_1^{j,p}}^\top B^\top(p_j, {h'}_j) N(p_i, \gamma_2^{i,p}) \\ &+ {\gamma_2^{j,p}}^\top B^\top(p_j, {h'}_j) N(p_i, \gamma_1^{i,p}) + {\gamma_2^{j,p}}^\top B^\top(\gamma_1^{j,p}, {h'}_j) A_i \end{aligned} \tag{4.134}$$

$$\begin{aligned} \left(\left({\Phi^{d_1}}_{q_i,q_j}\gamma_1\right)_{q_i,q_j}\gamma_2\right)_{{h'}_j} &= \Big((A_j {h'}_j)^\top B(\gamma_1^{i,p}, {h'}_i)\gamma_2^{i,p} + {\gamma_1^{j,p}}^\top B^\top(p_j, {h'}_j) B(p_i, {h'}_i)\gamma_2^{i,p} \\ &+ {\gamma_1^{i,p}}^\top B^\top(p_i, {h'}_i) B(p_j, {h'}_j)\gamma_2^{j,p} + (A_i {h'}_i)^\top B(\gamma_1^{j,p}, {h'}_j)\gamma_2^{j,p}\Big)_{{h'}_j} \\ &= {\gamma_2^{i,p}}^\top B^\top(\gamma_1^{i,p}, {h'}_i) A_j + {\gamma_2^{i,p}}^\top B^\top(p_i, {h'}_i) N(p_j, \gamma_1^{j,p}) \\ &+ {\gamma_1^{i,p}}^\top B^\top(p_i, {h'}_i) N(p_j, \gamma_2^{j,p}) + (A_i {h'}_i)^\top N(\gamma_1^{j,p}, \gamma_2^{j,p}) \end{aligned} \tag{4.135}$$

The dot-2 constraint function Φ^{d_2} of Eq. (4.58) depends on model parameters ${h'}_i$, ${s'}_i^P$, and ${s'}_j^P$. Derivatives ${\Phi^{d_2}}_u$, $\left({\Phi^{d_2}}_q\gamma\right)_u$, $\left({\Phi^{d_2}}_q{}^\top\zeta\right)_u$, and $\left(\left({\Phi^{d_2}}_q\gamma\right)_q\zeta\right)_u$, where γ

and ζ are constant known vectors and $u \in \{{h'}_i, {s'}_i^P, {s'}_j^P\}$, are [50]

$$\Phi^{d_2}_{{h'}_i} = {d_{i,j}}^\top A_i \tag{4.136}$$

$$\Phi^{d_2}_{{s'}_i^P} = (A_i {h'}_i)^\top (-A_i) = -{h'}_i^\top \tag{4.137}$$

$$\Phi^{d_2}_{{s'}_j^P} = (A_i {h'}_i)^\top (A_j) \tag{4.138}$$

$$\begin{aligned} (\Phi^{d_2}_{q_i,q_j}\gamma)_{{h'}_i} &= -{\gamma^{i,r}}^\top(A_i) + {d_{i,j}}^\top N(p_i, \gamma^{i,p}) - {\gamma^{i,p}}^\top B^\top(p_i, {s'}_i^P)A_i \\ &+ {\gamma^{j,r}}^\top A_i + {\gamma^{j,p}}^\top B^\top(p_j, {s'}_j^P)A_i \end{aligned} \tag{4.139}$$

$$(\Phi^{d_2}_{q_i,q_j}\gamma)_{{s'}_i^P} = -{\gamma^{i,p}}^\top B^\top(p_i, {h'}_i)A_i - (A_i {h'}_i)^\top N(p_i, \gamma^{i,p}) \tag{4.140}$$

$$\begin{aligned} (\Phi^{d_2}_{q_i,q_j}\gamma)_{{s'}_j^P} &= \left((A_i {h'}_i)^\top B(p_j, {s'}_j^P)\gamma^{j,p}\right)_{{s'}_j^P} + {\gamma^{i,p}}^\top B^\top(p_i, {h'}_i){d_{i,j}}_{{s'}_j^P} \\ &= (A_i {h'}_i)^\top N(p_j, \gamma^{j,p}) + {\gamma^{i,p}}^\top B^\top(p_i, {h'}_i)A_j \end{aligned} \tag{4.141}$$

$$(\Phi^{d_2}_{q_i,q_j}{}^{\top}\alpha)_{h'_i} = \alpha \begin{pmatrix} -A_i \\ C^{\top}(p_i, d_{i,j}) - B^{\top}(p_i, s'^P_i)A_i \\ A_i \\ B^{\top}(p_j, s'^P_j)A_i \end{pmatrix} \tag{4.142}$$

$$(\Phi^{d_2}_{q_i,q_j}{}^{\top}\alpha)_{s'^P_i} = \alpha \begin{pmatrix} 0 \\ -C^{\top}(p_i, A_i h'_i) - B^{\top}(p_i, h'_i)A_i \\ 0 \\ 0 \end{pmatrix} \tag{4.143}$$

$$(\Phi^{d_2}_{q_i,q_j}{}^{\top}\alpha)_{s'^P_j} = \alpha \begin{pmatrix} 0 \\ B^{\top}(p_i, h'_i)A_j \\ 0 \\ C^{\top}(p_j, A_i h'_i) \end{pmatrix} \tag{4.144}$$

$$\begin{aligned} ((\Phi^{d_2}{}_{q_i,q_j}\gamma_1)_{q_i,q_j}\gamma_2)_{h'_i} &= d_{i,j}{}^{\top}N(\gamma_1^{i,p}, \gamma_2^{i,p}) - \gamma_2^{i,r}{}^{\top}N(p_i, \gamma_1^{i,p}) - (\Phi^S_{q_i,q_j}\gamma_1)^{\top}N(p_i, \gamma_2^{i,p}) \\ &- \gamma_2^{i,p}{}^{\top}B^{\top}(\gamma_1^{i,p}, s'^P_i)A_i - \gamma_2^{i,p}{}^{\top}B^{\top}(p_i, s'^P_i)N(p_i, \gamma_1^{i,p}) \\ &+ \gamma_2^{j,r}{}^{\top}N(p_i, \gamma_1^{i,p}) + \gamma_2^{j,p}{}^{\top}B^{\top}(\gamma_1^{j,p}, s'^P_j)A_i \\ &+ \gamma_2^{j,p}{}^{\top}B^{\top}(p_j, s'^P_j)N(p_i, \gamma_1^{i,p}) \end{aligned} \tag{4.145}$$

$$\begin{aligned} ((\Phi^{d_2}{}_{q_i,q_j}\gamma_1)_{q_i,q_j}\gamma_2)_{s'^P_i} &= -\gamma_2^{i,p}{}^{\top}B^{\top}(p_i, h'_i)N(p_i, \gamma_1^{i,p}) \\ &+ (A_i h'_i)^{\top}N(\gamma_1^{i,p}, \gamma_2^{i,p}) + \gamma_2^{i,p}{}^{\top}B^{\top}(\gamma_1^{i,p}, h'_i)A_i \\ &+ \gamma_1^{i,p}{}^{\top}B^{\top}(p_i, h'_i)N(p_i, \gamma_2^{i,p}) \end{aligned} \tag{4.146}$$

$$\begin{aligned}\left(\left(\Phi^{d_2}{}_{q_i,q_j}\gamma_1\right)_{q_i,q_j}\gamma_2\right)_{s'^P_j} &= {\gamma_2^{i,p}}^\top B^\top(p_i,h'_i)N(p_j,\gamma_1^{j,p}) \\ &+ {\gamma_2^{i,p}}^\top B^\top(\gamma_1^{i,p},h'_i)A_j + (A_ih'_i)^\top N(\gamma_1^{j,p},\gamma_2^{j,p}) \\ &+ {\gamma_1^{i,p}}^\top B^\top(p_i,h'_i)N(p_j,\gamma_2^{j,p}) \end{aligned} \tag{4.147}$$

The distance constraint function Φ^D of Eq. (4.73) depends on model parameters C, ${s'}_i^P$, and ${s'}_j^P$. Derivatives ${\Phi^D}_u$, $\left({\Phi^D}_q\gamma\right)_u$, $\left({\Phi^D}_q{}^\top\zeta\right)_u$, and $\left(\left({\Phi^D}_q\gamma\right)_q\zeta\right)_u$, where γ and ζ are constant known vectors and $u \in \{C, {s'}_i^P, {s'}_j^P\}$, are [50]

$$\Phi^D_{{s'}_i^P} = -2{d_{i,j}}^\top A_i \tag{4.148}$$

$$\Phi^D_{{s'}_j^P} = 2{d_{i,j}}^\top A_j \tag{4.149}$$

$$\Phi^D_C = -2C \tag{4.150}$$

$$\begin{aligned}\left(\Phi^D_{q_i,q_j}\gamma\right)_{{s'}_i^P} = &- 2\left(\gamma^{j,r} + B(p_j,{s'}_j^P)\gamma^{j,p} - \gamma^{i,r} - B(p_i,{s'}_i^P)\gamma^{i,p}\right)^\top A_i \\ &- 2{d_{i,j}}^\top N(p_i,\gamma^{i,p}) \end{aligned} \tag{4.151}$$

$$\begin{aligned}\left(\Phi^D_{q_i,q_j}\gamma\right)_{{s'}_j^P} &= 2\left(\gamma^{j,r} + B(p_j,{s'}_j^P)\gamma^{j,p} - \gamma^{i,r} - B(p_i,{s'}_i^P)\gamma^{i,p}\right)^\top A_j \\ &+ 2{d_{i,j}}^\top N(p_j,\gamma^{j,p}) \end{aligned} \tag{4.152}$$

$$\left(\Phi^D_{q_i,q_j}\gamma\right)_C = 0 \tag{4.153}$$

$$\left(\Phi^D_{q_i,q_j}{}^\top\alpha\right)_{{s'}_i^P} = 2\alpha\begin{pmatrix} A_i \\ -C^\top(p_i,d_{i,j}) + B^\top(p_i,{s'}_i^P)A_i \\ -A_i \\ -B^\top(p_j,{s'}_j^P)A_i \end{pmatrix} \tag{4.154}$$

$$(\Phi^D_{q_i,q_j}{}^\top \alpha)_{s'^P_j} = 2\alpha \begin{pmatrix} A_j \\ -B^\top(p_i, s'^P_i)A_j \\ A_j \\ C^\top(p_j, d_{i,j}) + B^\top(p_j, s'^P_j)A_j \end{pmatrix} \tag{4.155}$$

$$(\Phi^D_{q_i,q_j}{}^\top \alpha)_C = 0 \tag{4.156}$$

$$\begin{aligned} ((\Phi^D{}_{q_i,q_j}\gamma_1)_{q_i,q_j}\gamma_2)_{s'^P_i} &= \gamma_1{}^\top \Phi^S_{q_i,q_j}{}^\top N(p_i, \gamma_2^{i,p}) + \gamma_2{}^\top \Phi^S_{q_i,q_j}{}^\top N(p_i, \gamma_1^{i,p}) \\ &- d_{i,j}{}^\top N(\gamma_1^{i,p}, \gamma_2^{i,p}) + \big(\gamma_2^{i,p\top} B^\top(\gamma_1^{i,p}, s'^P_i) \qquad (4.157) \\ &- \gamma_2^{j,p\top} B^\top(\gamma_1^{j,p}, s'^P_j)\big) A_i \qquad (4.158) \end{aligned}$$

$$\begin{aligned} ((\Phi^D{}_{q_i,q_j}\gamma_1)_{q_i,q_j}\gamma_2)_{s'^P_j} &= d_{i,j}{}^\top N(\gamma_1^{j,p}, \gamma_2^{j,p}) \\ &- \gamma_1{}^\top \Phi^S_{q_i,q_j}{}^\top N(p_j, \gamma_2^{j,p}) - \gamma_2{}^\top \Phi^S_{q_i,q_j}{}^\top N(p_j, \gamma_1^{j,p}) \\ &- \big(\gamma_2^{i,p\top} B^\top(\gamma_1^{i,p}, s'^P_i) - \gamma_2^{j,p\top} B^\top(\gamma_1^{j,p}, s'^P_j)\big) A_j \qquad (4.159) \end{aligned}$$

$$((\Phi^D{}_{q_i,q_j}\gamma_1)_{q_i,q_j}\gamma_2)_C = 0 \tag{4.160}$$

Consider vector function

$$\omega_0(a,k,j) = \begin{pmatrix} a_3 & 0 & a_1 \end{pmatrix}^\top \tag{4.161}$$

if $k = 3$ and $j = 1$,

$$\omega_0(a,k,j) = \begin{pmatrix} a_2 & a_1 & 0 \end{pmatrix}^\top \tag{4.162}$$

if $k = 2$ and $j = 1$,

$$\omega_0(a,k,j) = \begin{pmatrix} 0 & a_3 & a_2 \end{pmatrix}^\top \tag{4.163}$$

if $k = 3$ and $j = 2$,

$$\omega_0(a, k, j) = \begin{pmatrix} a_1 & 0 & 0 \end{pmatrix}^\top \tag{4.164}$$

if $k = 1$ and $j = 1$,

$$\omega_0(a, k, j) = \begin{pmatrix} 0 & a_2 & 0 \end{pmatrix}^\top \tag{4.165}$$

if $k = 2$ and $j = 2$,

$$\omega_0(a, k, j) = \begin{pmatrix} 0 & 0 & a_3 \end{pmatrix}^\top \tag{4.166}$$

if $k = 3$ and $j = 3$, where $a = \begin{pmatrix} a_1 & a_2 & a_3 \end{pmatrix}^\top$ is a three-dimensional vector and k,j are integers with properties $k, j \in \{1, 2, 3\}$ and $k \geq j$. Let $\gamma^i = \begin{pmatrix} \gamma^{i,r\top} & \gamma^{i,p\top} \end{pmatrix}^\top$ be a constant seven-dimensional vector with three-dimensional component $\gamma^{i,r}$ and four-dimensional component $\gamma^{i,p}$, and $(J'_i)_{kj}$ is the inertia tensor J'_i element positioned at row k and column j.

Consider the product of Eq. (A.136) of the Appendix of matrix M_i and vector γ^i. The derivative of product $(M_i\gamma^i)$ with respect to model parameter m_i is

$$(M_i\gamma^i)_{m_i} = \begin{pmatrix} \gamma^{i,r} - 2A_i\tilde{s'}_i^C G_i\gamma^{i,p} \\ 2{G_i}^\top\tilde{s'}_i^C {A_i}^\top\gamma^{i,r} \end{pmatrix} \tag{4.167}$$

In order to evaluate the derivative of $(M_i\gamma^i)$ with respect to model parameter ${s'}_i^C$, $(M_i\gamma^i)$ is re-written such that terms not depending on ${s'}_i^C$ are emphasized using the $\widehat{()}$ operator

$$M_i\gamma^i = \begin{pmatrix} (\widehat{m_i\gamma^{i,r}}) - 2(\widehat{m_iA(p_i)})\tilde{s'}_i^C(\widehat{G(p_i)\gamma^{i,p}}) \\ 2(\widehat{m_iG(p_i)}^\top)\tilde{s'}_i^C(\widehat{A(p_i)}^\top\gamma^{i,r}) + 4(G(p_i)^\top\widehat{J'_iG(p_i)}\gamma^{i,p}) \end{pmatrix} \tag{4.168}$$

Using the property of the $\sim$ operator that [27] $\tilde{a}b = -\tilde{b}a$ and canceling terms that do not depend on ${s'}_i^C$,

$$(M_i\gamma^i)_{{s'}_i^C} = 2m_i \begin{pmatrix} A_i(\widetilde{G_i\gamma^{i,p}}) \\ -{G_i}^\top(\widetilde{{A_i}^\top\gamma^{i,r}}) \end{pmatrix} \tag{4.169}$$

The product $(M_i\gamma^i)$ depends on inertia tensor elements $({J'}_i)_{kj}$ only through term $4G(p_i)^\top {J'}_i G(p_i)\gamma^{i,p}$. Therefore, the derivative of $(M_i\gamma^i)$ with respect to $({J'}_i)_{kj}$,$j \leq k$, is

$$(M_i\gamma^i)_{({J'}_i)_{kj}} = \begin{pmatrix} 0 \\ \\ 4{G_i}^\top({J'}_i)_{({J'}_i)_{kj}} G(p_i)\gamma^{i,p} \end{pmatrix} \tag{4.170}$$

If $j = k \in \{1,2,3\}$, then symmetric 3×3 matrix $({J'}_i)_{({J'}_i)_{kj}}$ has an element equal to one on k-th row and k-th column, all other elements being zero. If $j < k \in \{1,2,3\}$, then $({J'}_i)_{({J'}_i)_{kj}}$ has two elements equal to one, positioned at the k-th row and j-th column and the j-th row and k-th column, all other elements being zero. Therefore, the product of matrix $({J'}_i)_{({J'}_i)_{kk}}$ and three-dimensional vector $\zeta = \begin{pmatrix} \zeta_1 & \zeta_2 & \zeta_3 \end{pmatrix}^\top$ is

$$({J'}_i)_{({J'}_i)_{kk}}\zeta = \zeta_k = \omega_0(\zeta,k,k) \tag{4.171}$$

where function $\omega_0(\zeta,k,j)$, $j \leq k$, is defined in Eqs. (4.161) through (4.166). Product $({J'}_i)_{({J'}_i)_{kj}}\zeta$, $j < k$, is a vector having elements ζ_j and ζ_k in j and k positions, respectively, and zero as the remaining elements; i.e.,

$$({J'}_i)_{({J'}_i)_{kj}}\zeta = \omega_0(\zeta,k,j) \tag{4.172}$$

for $j < k \in \{1, 2, 3\}$. As a result, letting $\zeta = (G_i \gamma^{i,p})$,

$$(M_i \gamma^i)_{(J'_i)_{kj}} = \begin{pmatrix} 0 \\ \\ 4{G_i}^\top \omega_0((G_i \gamma^{i,p}), k, j) \end{pmatrix} \tag{4.173}$$

Consider the Coriolis term L_i of Eq. (A.128) of the Appendix,

$$L_i = \begin{pmatrix} L_{1,i} \\ \\ L_{2,i} \end{pmatrix} \tag{4.174}$$

where the upper-block of L_i is

$$L_{1,i} = 4 m_i E_i {G_i'}^\top G_i' {G_i}^\top {s'}_i^C \tag{4.175}$$

and the lower-block of L_i is

$$L_{2,i} = -8 {G_i'}^\top {J'}_i G_i {p_i}' \tag{4.176}$$

Since only the upper block $L_{1,i}$ depends on mass m_i, the derivative of L_i with respect to model parameter m_i is

$$(L_i)_{m_i} = \begin{pmatrix} 4 E_i {G_i'}^\top G_i' {G_i}^\top {s'}_i^C \\ \\ 0 \end{pmatrix} \tag{4.177}$$

Also, only the upper block $L_{1,i}$ depends on ${s'}_i^C$. Therefore, the derivative of L_i with respect to model parameter ${s'}_i^C$ is

$$(L_i)_{{s'}_i^C} = \begin{pmatrix} 4 m_i E_i {G_i'}^\top G_i' {G_i}^\top \\ \\ 0 \end{pmatrix} \tag{4.178}$$

Coriolis term L_i depends on inertia tensor elements $({J'}_i)_{kj}$ only through the lower-block $L_{2,i}$. Hence,

$$(L_i)_{(J'_i)_{kj}} = \begin{pmatrix} 0 \\ \\ -8 {G_i}^\top ({J'}_i)_{(J'_i)_{kj}} G_i {p_i}' \end{pmatrix} \tag{4.179}$$

Replacing $({J'}_i)_{({J'}_i)_{kj}}$ by its expression defined in Eqs. (4.171) and (4.171), in which $\zeta = (G_i {p_i}')$, the derivatives of L_i with respect to inertia tensor elements $({J'}_i)_{kj}$ are

$$(L_i)_{({J'}_i)_{kj}} = \begin{pmatrix} 0 \\ \\ -8{G_i}^{\top}\omega_0((G_i {p_i}'), k, j) \end{pmatrix} \tag{4.180}$$

Partial derivatives of the applied force component Q_i of Eq. (4.90) with respect to model parameters F_i^A and ${n'}_i^A$; TSDA parameters k, l_0, c, and $f(l, l')$; and RSDA parameters k_θ, c_θ, and $n(\theta, \theta')$ are defined by Eqs. (A.210) through (A.233) of the Appendix.

4.4 An Index-1 Formulation of the Adjoint DAE

Consider the functional

$$\Psi(\beta, t^2) = l(t^2, q^2, {q'}^2, \beta) + \int_{t^1}^{t^2} g(q, q', \lambda, \beta, t)dt \tag{4.181}$$

where final time t^2 is implicitly defined by the the condition [26]

$$\Omega(t^2, q(\beta, t^2), \beta) = 0 \tag{4.182}$$

The index-3 adjoint DAE of a multibody system with respect to the functional of Eq. (4.181) is defined [26] as

$$M\mu'' - Q^{\mu_v}\mu' - Q^{\mu_q}\mu + {\Phi_q}^{\top}\nu = s_a \tag{4.183}$$

$$\Phi_q\mu - r_a = 0 \tag{4.184}$$

where μ is the adjoint variable, ν is the adjoint Lagrange multiplier, and

$$Q^{\mu_q} = -\left[M'' + \left(\frac{d}{dt}(S_v^1)\right)^{\top} - (S_q^1)^{\top} + \left(({\Phi_q}^{\top}\lambda)_q{}^{\top}\right) + \left((Mv')_q\right)^{\top}\right] \tag{4.185}$$

$$Q^{\mu_v} = -(2M' + S_v^{1\top}) \tag{4.186}$$

$$s_a = g_q{}^\top - \frac{d}{dt}(g_{q'}{}^\top) \tag{4.187}$$

$$r_a = g_\lambda{}^\top \tag{4.188}$$

This system was shown in Chapter 3 to be stable in the backward direction. Differentiating the adjoint constraint of Eq. (4.184), the adjoint velocity constraint equation is obtained,

$$\Phi_q \mu' + (\Phi_q)' \mu - r_a{}' = 0 \tag{4.189}$$

Differentiating the velocity adjoint constraint of Eq. (4.189), the adjoint acceleration constraint equation is obtained,

$$\Phi_q \mu'' + 2(\Phi_q)' \mu' + (\Phi_q)'' \mu - r_a{}'' = 0 \tag{4.190}$$

Defining variables

$$\mu^v = \mu' \tag{4.191}$$

$$\mu^q = \mu \tag{4.192}$$

and the function

$$Q^\mu = Q^{\mu_v} \mu' + Q^{\mu_q} \mu = Q^{\mu_v} \mu^v + Q^{\mu_q} \mu^q \tag{4.193}$$

the equivalent index-1 DAE formulation [2], obtained by applying a stabilized index reduction [24] to the DAE of Eqs. (4.183) and (4.184), augmenting the resulting DAE with the constraint of Eq. (4.189), and introducing variables χ_a and ψ_a is

$$F_a(y, y', t) = \begin{pmatrix} {\mu'}^q - \mu^v + {\Phi_q}^\top {\chi_a}' \\ M{\mu'}^v + {\Phi_q}^\top {\psi_a}' - Q^\mu - s_a \\ \Phi_q \mu^v + (\Phi_q)' \mu^q - {r_a}' \\ \Phi_q \mu^q - r_a \end{pmatrix} = 0 \tag{4.194}$$

where

$$y = \begin{pmatrix} \chi_a \\ \psi_a \\ \mu^v \\ \mu^q \end{pmatrix} \tag{4.195}$$

A variable step-size BDF integration method of up to order six converges for the index-1 DAE of Eq. (4.194), as is shown in Section 4.5, as long as integration is performed in the backward direction. The Jacobian of the non-linear equation that results from applying a BDF integration method to the index-1 DAE of Eq. (4.194) is

$$J_a = \begin{pmatrix} \alpha_1 {\Phi_q}^\top & 0 & -I & \alpha_1 I \\ 0 & \alpha_1 {\Phi_q}^\top & \alpha_1 M - Q^{\mu^v} & -Q^{\mu^q} \\ 0 & 0 & \Phi_q & (\Phi_q)' \\ 0 & 0 & 0 & \Phi_q \end{pmatrix} \tag{4.196}$$

Evaluations of function F_a, Jacobian J_a, functions ${r_a}'$, ${r_a}''$, and consistent initial conditions require computation of the following derivatives:

1. The position constraint Jacobian Φ_q; derivatives of the form $\left({\Phi_q}^\top \zeta\right)_q$, $\left(M\gamma\right)_q$, S_q^1; and S_v^1. The analytic evaluation of these derivatives is shown in Section 4.2.

2. First and second derivatives of mass matrix M and derivative of matrix S_v^1.

3. Derivatives of function g, g_q, $(g_{q'})'$, g_λ, $(g_\lambda)'$, and $(g_\lambda)''$.

4.4.1 Consistent Initial Conditions

Consistent initial conditions for the direct differentiation DAE of Eqs. (4.183) and (4.184) are computed after the final time t^2 is reached and are given by the following conditions [26]:

$$M^S(t^2)\begin{pmatrix} \mu(t^2) \\ \\ \eta(t^2) \end{pmatrix} = \begin{pmatrix} {l_{q'}(t^2)}^\top \\ \\ {g_\lambda}^\top(t^2) \end{pmatrix} \tag{4.197}$$

where $M^S(t^2)$ is the non-singular Schur-complement matrix defined by Eq. (4.25) at final time t^2. As a result of solving the linear system of Eq. (4.197), the adjoint variable $\mu(t^2)$ is available. In order to obtain the derivative $\mu'(t^2)$ of the adjoint variable at the final time, the following linear system must be solved [26]:

$$\begin{aligned} M^S(t^2)\begin{pmatrix} \mu'(t^2) \\ -\gamma(t^2) \end{pmatrix} = & - \begin{pmatrix} (M'\mu + {S_{q'}^1}^\top\mu - {(\Phi_q q')_q}^\top\eta + {l_q}^\top + {g_{q'}}^\top)(t^2) \\ {\Phi_q}'\mu(t^2) - {g_\lambda}'(t^2) \end{pmatrix} \\ & + \begin{pmatrix} {\Omega_q}^\top(t^2) \\ 0 \end{pmatrix}\xi(t^2) \end{aligned} \tag{4.198}$$

where scalar $\xi(t^2)$ is defined [26] as

$$\begin{aligned} \xi(t^2) = & - \frac{\Big(q'(t^2)^\top {(\Phi_q q')_q}^\top(t^2) + q''(t^2)^\top {\Phi_q}^\top(t^2)\Big)\eta(t^2)}{\Omega_t(t^2) + (\Omega_q q')(t^2)} \\ & + \frac{\Big(\frac{d}{dt}l(t,q,q')\Big)(t^2) + g(t^2)}{\Omega_t(t^2) + (\Omega_q q')(t^2)} \end{aligned} \tag{4.199}$$

The adjoint differential equation of Eq. (4.183), together with adjoint acceleration constraint of Eq. (4.190), at the final time, form the linear system

$$M^S(t^2)w_2^a = b_2^a \tag{4.200}$$

where w_2^a is the vector of unknowns,

$$w_2^a = \begin{pmatrix} \mu''(t^2)^\top & \nu(t^2)^\top \end{pmatrix}^\top$$

and

$$b_2^a = \begin{pmatrix} (s_a + Q^{\mu_v}\mu' + Q^{\mu_q}\mu)(t^2) \\ (r_a{}'' - 2(\Phi_q)'\mu' - (\Phi_q)''\mu)(t^2) \end{pmatrix}$$

which solves for the second derivative $\mu''(t^2)$ of the adjoint variable and for the adjoint Lagrange multipliers $\nu(t^2)$ at the final time. As a result, the values of y and y' at final time t^2, which represent initial conditions for the index-1 DAE of Eq. (4.194), are

$$y(t^2) = \begin{pmatrix} \chi_a(t^2) \\ \psi_a(t^2) \\ \mu'(t^2) \\ \mu(t^2) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \mu'(t^2) \\ \mu(t^2) \end{pmatrix} \tag{4.201}$$

and

$$y'(t^2) = \begin{pmatrix} \chi_a{}' \\ \psi_a{}' \\ \mu''(t^2) \\ \mu'(t^2) \end{pmatrix} = \begin{pmatrix} 0 \\ \nu(t^2) \\ \mu''(t^2) \\ \mu'(t^2) \end{pmatrix} \tag{4.202}$$

4.4.2 Derivatives of Mass Matrix and of Matrix S_v^1

Substituting for $x \equiv q_i$ and $X \equiv M_i$ into Eq. (A.7), of the Appendix,

$$M_i{}' = \begin{pmatrix} (M_i u_{1,0})_{q_i} q_i{}' + (M_i u_{1,0})_t & \dots & (M_i u_{7,0})_{q_i} q_i{}' + (M_i u_{7,0})_t \end{pmatrix} \tag{4.203}$$

where $u_{l,0}$, $l = 1, 2, \dots, 7$, is the seven-dimensional unit vector with all elements zero except the l-th, which is one. Each column of $M_i{}'$ is independently evaluated using the identity of Eq. (4.85). For a multibody system with constant mass and moments of inertia, partial derivatives of mass elements with respect to time, of the form $(M_i u_{l,0})_t$, $l = 1, 2, \dots, 7$, are zero.

The second derivative $M_i{}''$ is obtained by substituting for $x \equiv q_i$ and $X \equiv M_i$ into Eq. (A.12) of the Appendix,

$$M_i{}'' = \begin{pmatrix} (m_{i,1})'' & \dots & (m_{i,7})'' \end{pmatrix} \tag{4.204}$$

where $m_{i,l} = M_i u_{l,0}$, $l = 1, 2, \dots, 7$, are the columns of M_i. Derivatives $(m_{i,l})''$ are obtained by applying the identity of Eq. (A.13) of the Appendix to columns $m_{i,l}$, $l = 1, 2, \dots, 7$,

$$(m_{i,l})'' = ((M_i u_{l,0})_{q_i} q_i{}')_{q_i} + (M_i u_{l,0})_{q_i} q_i{}'' + 2(M_i u_{l,0})_{q_i,t} q_i{}' + (M_i u_{l,0})_{t,t} \tag{4.205}$$

where partial derivatives $(M_i u_{l,0})_{q_i,t}$ and $(M_i u_{l,0})_{t,t}$ are zero for multibody systems with constant mass and moments of inertia.

Consequently, the first derivative of the mass matrix is assembled as

$$M' = (diag(M_i))' = diag((M_i)', i = 1, 2, \dots, n_b) \tag{4.206}$$

where $(M_i)'$ is defined by Eq. (4.203), and the second derivative of the mass matrix is

$$M'' = (diag(M_i))'' = diag((M_i)'', i = 1, 2, \ldots, n_b) \tag{4.207}$$

where $(M_i)''$ is defined by Eq. (4.204). Hence, derivatives of the mass matrix are obtained by independently evaluating the derivatives of each block M_i, using Eqs. (4.203) and (4.204). Derivatives of the form $(M_i\gamma^i)_{q_i}$ and $((M_i\gamma_1^i)_{q_i}\gamma_2^i)_{q_i}$; where γ^i, γ_1^i, and γ_2^i are 7-dimensional constant vectors; are presented in Section A.7 of the Appendix.

Derivatives of matrix blocks $L_{iq_i'}$ are obtained by substituting for $x \equiv \begin{pmatrix} {q_i}^\top & {q_i'}^\top \end{pmatrix}^\top$ and $X \equiv L_{iq_i'}$ into the identity of Eq. (A.7) of the Appendix,

$$(L_{iq_i'})' = \begin{pmatrix} (L_{iq_i'})_1{}' & \ldots & (L_{iq_i'})_7{}' \end{pmatrix} \tag{4.208}$$

where the derivative of the l-th column, $(L_{iq_i'})_l$, $l = 1, 2, \ldots, 7$, of matrix $(L_{iq_i'})$ is

$$(L_{iq_i'})_l{}' = (L_{iq_i'}u_{l,0})_{q_i}q_i' + (L_{iq_i'}u_{l,0})_{q_i'}q_i'' + (L_{iq_i'}u_{l,0})_t \tag{4.209}$$

Derivatives of the form $(L_{iq_i'}\gamma^i)_{q_i}$ and $(L_{iq_i'}\gamma^i)_{q_i'}$ are defined by Eqs. (A.160) and (A.161), respectively, in the Appendix. It should be noted that $(L_{iq_i'}u_{l,0})_t = 0$, $l = 1, 2, \ldots, 7$, for a multibody system with constant mass and moments of inertia.

For an applied force Q_i acting on body i, derivatives of matrix blocks $Q_{iq_i'}$ are obtained by substituting for $x \equiv \begin{pmatrix} {q_i}^\top & {q_i'}^\top \end{pmatrix}^\top$ and $X \equiv Q_{iq_i'}$ into the identity of Eq. (A.7) of the Appendix,

$$(Q_{iq_i'})' = \begin{pmatrix} (Q_{iq_i'})_1{}' & \ldots & (Q_{iq_i'})_7{}' \end{pmatrix} \tag{4.210}$$

where the derivative of the l-th column, $(Q_{iq_i'})_l$, $l = 1, 2, \ldots, 7$, of matrix $(Q_{iq_i'})$ is

$$(Q_{iq_i'})_l{}' = (Q_{iq_i'u_{l,0}})_{q_i} q_i{}' + (Q_{iq_i'u_{l,0}})_{q_i'} q_i{}'' + (Q_{iq_i'u_{l,0}})_t \tag{4.211}$$

For a TSDA or RSDA force Q_i acting between bodies i and j, derivatives $(Q_{iq'_{ij}})'$, where $q_{ij} = \begin{pmatrix} q_i{}^\top & q_j{}^\top \end{pmatrix}^\top$, are obtained by substituting for $x \equiv \begin{pmatrix} q_{ij}{}^\top & q'_{ij}{}^\top \end{pmatrix}^\top$ and $X \equiv Q_{iq'_{ij}}$ into the identity of Eq. (A.7) of the Appendix,

$$(Q_{iq'_{ij}})' = \begin{pmatrix} (Q_{iq'_{ij}})_1{}' & \ldots & (Q_{iq'_{ij}})_{14}{}' \end{pmatrix} \tag{4.212}$$

where the derivative of the l-th column, $(Q_{iq'_{ij}})_l$, $l = 1, 2, \ldots, 14$, of matrix $(Q_{iq'_{ij}})$ is

$$(Q_{iq'_{ij}})_l{}' = (Q_{iq'_{ij}\eta_{l,0}})_{q_{ij}} q'_{ij} + (Q_{iq'_{ij}\eta_{l,0}})_{q'_{ij}} q''_{ij} + (Q_{iq'_{ij}\eta_{l,0}})_t \tag{4.213}$$

where $\eta_{l,0}$, $l = 1, 2, \ldots, 14$, is the 14-dimensional unit vector with all elements zero except the l-th, which is one. Derivatives of the form $(Q_{iq'_{ij}}\gamma^{i,j})_{q_{ij}}$ where $\gamma^{i,j}$ is a constant 14-dimensional vector, are defined in the Appendix by Eqs. (A.176) and (A.177), respectively, for TSDA and Eqs. (A.199) and (A.200), respectively, for RSDA. Derivatives of the form $(Q_{iq'_{ij}}\gamma^{i,j})_{q'_{ij}}$ are defined in the Appendix by Eqs. (A.178) and (A.179), respectively, for TSDA and Eqs. (A.201) and (A.202), respectively, for RSDA. Therefore,

$$(S_v^1)' = diag((L_{iq_i'})', i = 1, 2, \ldots, n_b) + Q_{q'}{}' \tag{4.214}$$

is obtained by independently evaluating the derivatives $(L_{iq_i'})'$ and $(Q_{iq_l'})'$, using Eqs. (4.208), (4.210), and (4.212).

Derivatives g_q, $(g_{q'})'$, g_λ, $(g_\lambda)'$, and $(g_\lambda)''$ are assumed to be given separately.

They depend on the actual optimization or optimal control problem for which gradients are computed [50].

4.5 Existence and Uniqueness of the Solution of Index-1 DAE Formulations of Motion, Sensitivity, and Adjoint Equations

The index-1 DAE formulations of motion, sensitivity, and adjoint equations of Eqs. (4.20), (4.113), and (4.194) have the form

$$F(y, y', t) = \begin{pmatrix} {y_4}' - y_3 + {\Phi_q}^\top {y_1}' \\ M{y_3}' + {\Phi_q}^\top {y_2}' + f_2(y_3, y_4) \\ \Phi_q y_3 + f_3(y_4) \\ f_4(y_4) \end{pmatrix} = 0 \tag{4.215}$$

where $y = \begin{pmatrix} {y_1}^\top & {y_2}^\top & {y_3}^\top & {y_4}^\top \end{pmatrix}^\top$. For the index-1 formulation of the equations of motion of Eq. (4.20),

$$\begin{aligned} y_1 &= \chi \\ y_2 &= \psi \\ y_3 &= v \equiv q' \\ y_4 &= q \\ f_2(y_3, y_4) &= -S^1(q, q', \beta, t) \\ f_3(y_4) &= \Phi_t(q, \beta, t) \\ f_4(y_4) &= \Phi(q, \beta, t) \end{aligned}$$

For the direct differentiation sensitivity index-1 formulation of Eq. (4.113),

$$\begin{aligned} y_1 &= \chi_d \\ y_2 &= \psi_d \\ y_3 &= x^v \equiv {q_\beta}' \\ y_4 &= x^q \equiv q_\beta \\ f_2(y_3, y_4) &= -Q^x(y_3, y_4) - s_d(q, q', q'', \beta, t) \\ f_3(y_4) &= (\Phi_q)' y_4 - r_d(q, \beta, t)' \\ f_4(y_4) &= \Phi_q y_4 - r_d(q, \beta, t) \end{aligned}$$

where Q^x, s_d, and r_d are defined by Eqs. (4.112), (4.104), and (4.105). For the adjoint index-1 formulation of Eq. (4.194),

$$\begin{aligned} y_1 &= \chi_a \\ y_2 &= \psi_a \\ y_3 &= \mu^v \equiv \mu' \\ y_4 &= \mu^q \equiv \mu \\ f_2(y_3, y_4) &= -Q^\mu(y_3, y_4) - s_a(q, q', \lambda, \beta, t) \\ f_3(y_4) &= (\Phi_q)' y_4 - r_a(q, q', \lambda, \beta, t)' \\ f_4(y_4) &= \Phi_q y_4 - r_a(q, q', \lambda, \beta, t) \end{aligned}$$

where Q^μ, s_a, and r_a are defined by Eqs. (4.193), (4.187), and (4.188). It should be noted that for all three formulations, $f_{4_{y_4}} = \Phi_q$. Also, in the index-1 formulation of the equations of motion, matrices M and Φ_q depend on $y_4 \equiv q$, while in the index-1 formulations of direct differentiation and adjoint equations, they do not depend on corresponding state vectors $y = \begin{pmatrix} \chi_d{}^\top & \psi_d{}^\top & x^{v\top} & x^{q\top} \end{pmatrix}^\top$ and $y = \begin{pmatrix} \chi_a{}^\top & \psi_a{}^\top & \mu^{v\top} & \mu^{q\top} \end{pmatrix}^\top$. The theorems that follow are proved for later use. They will be used in the proof of existence and uniqueness of a BDF integration method applied to the index-1 formulation of Eq. (4.20) for the equations of motion, the index-1 formulation of Eq. (4.113) for direct differentiation sensitivity equations, and the index-1 formulation of Eq. (4.194) for adjoint equations.

Theorem 4.1. *Matrix $J'^C = J' + m\tilde{s}'^{C^2}$ is positive definite, where J' is the positive definite inertia tensor of the body, m is the mass, and s'^C is the position of the center*

of mass with respect to the body-fixed coordinate frame.

Proof. The definition of the inertia tensor J' for a rigid body is [27]

$$J' = -\int_m \tilde{s'}^P \tilde{s'}^P dm(P) \tag{4.216}$$

where s'^P is the position of a point P in the rigid body with respect to the body-fixed coordinate frame, $dm(P)$ is the differential unit of mass at point P, and integration is performed over the entire mass m of the body. Operator $\sim$, applied to a vector $a = \begin{pmatrix} a_x & a_y & a_z \end{pmatrix}^\top$, is [27]

$$\tilde{a} = \begin{pmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{pmatrix}$$

with the properties that $\tilde{a}a = 0$, $\tilde{a}^\top = -\tilde{a}$, and $\det(\tilde{a}) = 0$. The definition of the position of the center of mass s'^C is [27]

$$\tilde{s'}^C = \frac{1}{m}\int_m \tilde{s'}^P dm(P) \tag{4.217}$$

Therefore,

$$J'^C = J' + m\tilde{s'}^{c^2} = -\int_m \tilde{s'}^P \tilde{s'}^P dm(P) + \frac{1}{m}\left(\int_m \tilde{s'}^P dm(P)\right)^2 \tag{4.218}$$

where matrix J'^C is symmetric,

$$\begin{aligned} J'^{C\top} &= J'^\top + m(\tilde{s'}^C \tilde{s'}^C)^\top = J' + m(\tilde{s'}^C)^\top (\tilde{s'}^C)^\top \\ &= J' + m(-\tilde{s'}^C)(-\tilde{s'}^C) = J' + m(-\tilde{s'}^C)^2 = J'^C \end{aligned} \tag{4.219}$$

Let a be a three-dimensional constant vector and $j'^C(a)$ the product

$$\begin{aligned} j'^C(a) &= a^\top J'^C a = a^\top (J' + m\tilde{s'^c}^2)a = -\int_m a^\top \tilde{s'}^P \tilde{s'}^P a \cdot dm(P) \\ &+ \frac{1}{m}\left(\int_m a^\top \tilde{s'}^P dm(P)\right)\cdot\left(\int_m \tilde{s'}^P a \cdot dm(P)\right) \end{aligned} \tag{4.220}$$

Using the identity $(\tilde{s'}^P)^\top = -\tilde{s'}^P$, $j'^C(a)$ is re-written as

$$\begin{aligned} j'^C(a) &= \int_m a^\top (\tilde{s'}^P)^\top \tilde{s'}^P a \cdot dm(P) - \frac{1}{m}\left(\int_m \tilde{s'}^P a \cdot dm(P)\right)^\top \left(\int_m \tilde{s'}^P a \cdot dm(P)\right) \\ &= \int_m \|\tilde{s'}^P a\|_2^2 dm(P) - \frac{1}{m}\|s^a\|_2^2 \end{aligned} \tag{4.221}$$

where $s^a = \int_m \tilde{s'}^P a \cdot dm(P)$.

Consider the vector function $f(x) : \Omega \subset \mathbb{R}^N \rightarrow \mathbb{R}^N$,

$$f(x) = \begin{pmatrix} f_1(x) & \dots & f_N(x) \end{pmatrix}^\top \tag{4.222}$$

where

$$f_i : \Omega \rightarrow \mathbb{R}, i = 1, 2, \dots, N \tag{4.223}$$

The absolute value of the scalar product of function $f_i(x)$ with function $u : \Omega \rightarrow \mathbb{R}$

$$u(x) = 1 \tag{4.224}$$

is, according to the Cauchy-Schwarz inequality [9], bounded by

$$|(f_i, u)_\Omega| \leq \|f_i\|_\Omega \cdot \|u\|_\Omega \tag{4.225}$$

where [9]

$$(f_i, u)_\Omega \equiv \int_\Omega f_i(x) u(x) dx \tag{4.226}$$

$$\|f_i\|_\Omega \equiv \sqrt{(f_i, f_i)_\Omega} \tag{4.227}$$

As a result, the inequality of Eq. (4.225) is

$$|\int_{\Omega} f_i(x)u(x)dx| \leq \sqrt{\int_{\Omega} f_i^2(x)dx} \cdot \sqrt{\int_{\Omega} u^2(x)dx} \tag{4.228}$$

which, by squaring terms and accounting for the definition $u(x) = 1$, is re-written as

$$\left(\int_{\Omega} f_i(x)dx\right)^2 \leq \int_{\Omega} 1dx \int f_i^2(x)dx \tag{4.229}$$

Since $f(x) = \begin{pmatrix} f_1(x) & \dots & f_N(x) \end{pmatrix}^{\top}$

$$\int_{\Omega} f^{\top}(x)f(x)dx = \int_{\Omega} \sum_{i=1}^{N} f_i^2(x)dx = \sum_{i=1}^{N} \int_{\Omega} f_i^2(x)dx \tag{4.230}$$

Pre-multiplying Eq. (4.230) by positive $\int_{\Omega} 1dx$, where sub-domain Ω is assumed to have non-zero measure [9],

$$\left(\int_{\Omega} 1dx\right) \cdot \left(\int_{\Omega} f^{\top}(x)f(x)dx\right) = \sum_{i=1}^{N} \left(\int_{\Omega} 1dx\right) \cdot \left(\int_{\Omega} f_i^2(x)dx\right) \tag{4.231}$$

Using the inequality of Eq. (4.229),

$$\begin{aligned} \left(\int_{\Omega} 1dx\right) \cdot \left(\int_{\Omega} f^{\top}(x)f(x)dx\right) &= \sum_{i=1}^{N} \left(\int_{\Omega} 1dx\right) \cdot \left(\int_{\Omega} f_i^2(x)dx\right) \\ &\geq \sum_{i=1}^{N} \left(\int_{\Omega} f_i(x)dx\right)^2 \end{aligned} \tag{4.232}$$

Let $\omega = \int_{\Omega} f(x)dx = \begin{pmatrix} \int_{\Omega} f_1(x)dx & \dots & \int_{\Omega} f_N(x)dx \end{pmatrix}^{\top}$. Then, the euclidian norm of vector ω is

$$\|\omega\|_2 = \left(\int_{\Omega} f(x)dx\right)^{\top} \cdot \left(\int_{\Omega} f(x)dx\right) = \sum_{i=1}^{N} \left(\int_{\Omega} f_i(x)dx\right)^2 \tag{4.233}$$

Substituting Eq. (4.233) into the inequality of Eq. (4.232),

$$\left(\int_{\Omega} f(x)dx\right)^{\top} \cdot \left(\int_{\Omega} f(x)dx\right) \leq \left(\int_{\Omega} 1dx\right) \cdot \left(\int_{\Omega} f^{\top}(x)f(x)dx\right) \tag{4.234}$$

and dividing by the strictly positive scalar $\int_\Omega 1dx$, the following inequality is obtained:

$$\left(\int_\Omega f^\top(x)f(x)dx\right) - \frac{1}{\int_\Omega 1dx}\left(\int_\Omega f(x)dx\right)^\top \left(\int_\Omega f(x)dx\right) \geq 0 \tag{4.235}$$

Taking $N = 3, x = m(P), \Omega = m$, and $f(x) = \tilde{s'}^P a$, it follows that $\int_\Omega 1{\cdot}dx = \int_m 1{\cdot}dm(P) = m$ and the inequality of Eq. (4.235) is re-written as

$$\int_\Omega (\tilde{s}^P a)^\top (\tilde{s}^P a)dm(P) - \frac{1}{m}\|s^a\|_2 \geq 0 \tag{4.236}$$

so $j'^C(a) = a^\top J'^C a \geq 0$, for an arbitrary vector $a{\in}\mathbb{R}^3$. In order for the inequality of Eq. (4.235) to become an equality, it is necessary [9] for $f(x)$ to be constant; i.e., $f(x) = c{\in}\mathbb{R}^N$. Therefore, $\tilde{s'}^P a = c{\in}\mathbb{R}^3$ for any $s'^P(m(P))$; i.e., being given an arbitrary vector a, the vector product

$$\tilde{s'}^P(m(P))a = c \tag{4.237}$$

needs to be constant for any point P inside the body, in order to obtain equality in Eq. (4.236). As a result, Eq. (4.237) must hold for $s^P{\in}\{i', j', k'\}$, where $\{i', j', k'\}$ are the unit vectors

$$i' = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}^\top \tag{4.238}$$

$$j' = \begin{pmatrix} 0 & 1 & 0 \end{pmatrix}^\top \tag{4.239}$$

$$k' = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^\top \tag{4.240}$$

Consequently,

$$\tilde{i'}a = c \tag{4.241}$$

$$\tilde{j'}a = c \tag{4.242}$$

$$\tilde{k'}a = c \tag{4.243}$$

where a is the arbitrary given vector of Eq. (4.220). Equations (4.241) through (4.243) can only hold if $c = 0$. Thus, vector a must be zero. As a result,

$$j'^{C}(a) = a^{\top} J'^{C} a = 0$$

only if $a = 0$. Otherwise $j'^{C}(a) > 0$, as shown in Eq.(4.236). Therefore, J'^{C} is positive definite. ■

Theorem 4.2. *The rank of the matrix*

$$G(p) = \begin{pmatrix} -e & -\tilde{e} + e_0 I \end{pmatrix} \tag{4.244}$$

is three for any normalized Euler parameter vector $p = \begin{pmatrix} e_0 & e^{\top} \end{pmatrix}$.

Proof. For the 3×4 matrix G, the following identity holds [27]:

$$GG^{\top} = I_3 \tag{4.245}$$

where I_3 is the 3×3 identity matrix. Applying Sylvester's inequality [23] to the identity of Eq. (4.245),

$$\begin{aligned} rank(G^{\top}) + rank(G) - 4 &\leq rank(GG^{\top}) \\ &\leq min\big(rank(G^{\top}), rank(G)\big) \\ &= rank(G) \end{aligned} \tag{4.246}$$

it follows that

$$rank(G) \geq rank(GG^{\top}) = rank(I_3) = 3 \tag{4.247}$$

Since G has three rows, its rank cannot exceed three. Therefore, $rank(G) = 3$. ■

Theorem 4.3. *If J is a 3×3 non-singular matrix, then the 4×4 matrix $J^G = G^\top(p)JG(p)$ has rank three, for any normalized Euler parameter vector p.*

Proof. According to Sylvester's inequality [23],

$$\begin{aligned} 3 &= rank(G^\top) + rank(J) - 3 \leq rank(G^\top J) \\ &\leq min\big(rank(G^\top), rank(J)\big) = 3 \end{aligned}$$

Therefore, $rank(G^\top J) = 3$. Applying Sylvester's inequality to the product $G^\top JG$, it follows that

$$\begin{aligned} 3 &= rank(G^\top J) + rank(G) - 3 \leq rank(G^\top JG) \\ &\leq min\big(rank(G^\top J), rank(G)\big) = 3 \end{aligned}$$

Consequently, $rank(G^\top JG) = 3$ ■

Corollary 4.4. *The null space of matrix $J^G = G^\top(p)JG(p)$, with J non-singular, has dimension one and is*

$$\mathcal{N}(G^\top(p)JG(p)) = \{p\}$$

where p is the normalized Euler parameter vector.

Proof. The following equality holds [23]:

$$rank(J^G) + dim\big(\mathcal{N}(J^G)\big) = 4$$

where $dim\big(\mathcal{N}(J^G)\big)$ represents the dimension of the null space of J^G. Therefore,

$$dim\big(\mathcal{N}(J^G)\big) = 4 - rank(J^G) = 1 \tag{4.248}$$

Since [27]

$$G(p)p = 0 \tag{4.249}$$

it follows that

$$J^G p = G^\top(p) J G(p) p = 0 \tag{4.250}$$

Therefore,

$$p \in \mathcal{N}(J^G)$$

and because $dim(\mathcal{N}(J^G)) = 1$,

$$\mathcal{N}(J^G) = \{p\}$$

■

Theorem 4.5. *Matrix*

$$M_i(p_i) \begin{pmatrix} m_i I_3 & -2m_i A(p_i) \tilde{s'}_i^C G(p_i) \\ 2m_i G^\top(p_i) \tilde{s'}_i^C A^\top(p_i) & 4G^\top(p_i) J'_i G(p_i) \end{pmatrix}$$

is a 7×7 *matrix of rank six, where* p_i *is the normalized Euler parameter vector of body* i, $A(p_i)$ *is the orientation matrix [27],* $G(p_i)$ *is the matrix defined by Eq.* (4.244), m_i *is the mass of body* i, s'^C_i *is the position of the center of mass of body* i *with respect to the body fixed coordinate frame, and* I_3 *is the* 3×3 *identity matrix.*

Proof. Consider an arbitrary seven-dimensional vector

$$v = \begin{pmatrix} v^{r\top} & v^{p\top} \end{pmatrix}$$

with tree-dimensional component v^r and four-dimensional component v^p. For vector

$$\omega_i = \begin{pmatrix} 0 & {p_i}^\top \end{pmatrix} \in \mathcal{N}(M_i(p_i)),$$

$$\begin{aligned} M_i(p_i)\omega_i &= \begin{pmatrix} m_i I_3 & -2m_i A(p_i)\tilde{s'}_i^C G(p_i) \\ 2m_i G^\top(p_i)\tilde{s'}_i^C A^\top(p_i) & 4G^\top(p_i)J_i' G(p_i) \end{pmatrix} \begin{pmatrix} 0 \\ p_i \end{pmatrix} \\ &= \begin{pmatrix} -2m_i A(p_i)\tilde{s'}_i^C G(p_i)p_i \\ 4G^\top(p_i)J_i' G(p_i)p_i \end{pmatrix} = 0 \end{aligned}$$

since $G(p_i)p_i = 0$ [27]. Assume that $v \in \mathcal{N}(M_i(p_i))$; i.e., $M_i(p_i)v = 0$. Then,

$$m_i v^r - 2m_i A(p_i)\tilde{s'}_i^C G(p_i)v^p = 0 \tag{4.251}$$

$$2m_i G^\top(p_i)\tilde{s'}_i^C A^\top(p_i)v^r + 4G^\top(p_i)J_i' G(p_i)v^p = 0 \tag{4.252}$$

Therefore, Eq. (4.251) states that

$$v^r = 2A(p_i)\tilde{s'}_i^C G(p_i)v^p \tag{4.253}$$

which, substituted into Eq. (4.252), yields

$$4m_i G^\top(p_i)\tilde{s'}_i^C A^\top(p_i)A(p_i)\tilde{s'}_i^C G(p_i)v^p + 4G^\top(p_i)J_i' G(p_i)v^p = 0 \tag{4.254}$$

Since $A^\top(p_i)A(p_i) = I_3$, Eq. (4.254) is re-written as

$$\big(m_i G^\top(p_i)\tilde{s'}_i^{c^2} G(p_i) + G^\top(p_i)J_i' G(p_i)\big)v^p = 0 \tag{4.255}$$

Factoring terms,

$$G^\top(p_i)(J_i' + m_i\tilde{s'}_i^{c^2})G(p_i)v^p = 0 \tag{4.256}$$

where matrix $J_i'^G = J_i' + m_i\tilde{s'}_i^{c^2}$ is non-singular, according to Theorem 4.1. As a result of Corollary 4.4, the null-space of matrix $G^\top(p_i)J_i'^G G(p_i)$ is

$$\mathcal{N}(G^\top(p_i)J_i'^G G(p_i)) = \{p_i\}$$

Therefore, $v^p = \alpha p_i$, where α is a non-zero scalar. Using Eq. (4.253),

$$v^r = 2A(p_i)\tilde{s}_i^C G(p_i)\alpha p_i = 0$$

so $v = \begin{pmatrix} v^{r\top} & v^{p\top} \end{pmatrix} = \begin{pmatrix} 0 & \alpha {p_i}^\top \end{pmatrix}$. Thus, if $v \in \mathcal{N}(M_i(p_i))$, it has the form

$$v = \begin{pmatrix} 0 \\ \\ \alpha p_i \end{pmatrix}$$

Therefore, the null-space of matrix $\mathcal{N}(M_i(p_i))$ consists of only one vector,

$$\mathcal{N}(M_i(p_i)) = \left\{ \begin{pmatrix} 0 \\ \\ p_i \end{pmatrix} \right\} \tag{4.257}$$

and $dim(\mathcal{N}(M_i(p_i))) = 1$. Since $(M_i(p_i))$ is a 7×7 matrix, it follows that

$$rank(M_i(p_i)) = 7 - dim(\mathcal{N}(M_i(p_i))) = 6$$

■

As a result, matrix $M_i(p_i)$ has six linearly independent rows(columns), for any normalized four-dimensional vector p_i. Thus, the multibody system mass matrix

$$M(q) = diag(M_i(q_i), i = 1, 2, \ldots, n_b)$$

has $6n_b$ linearly independent rows(columns), for any consistent generalized coordinate $q = \begin{pmatrix} {q_1}^\top & \ldots & {q_{n_b}}^\top \end{pmatrix}^\top$; i.e., any $q(\beta, t)$ such that $\Phi(q, \beta, t) = 0$. Therefore,

$$M^I(q) = \begin{pmatrix} I_n & 0 \\ \\ 0 & M(q) \end{pmatrix} \tag{4.258}$$

where I_n is the n-dimensional identity matrix, has $n + 6n_b$ linearly independent rows(columns) for any consistent q. Hence, the theorem that follows is proved.

Theorem 4.6. *The rank of mass matrix $M^I(q)$ is*

$$rank(M^I(q)) = n + 6n_b \tag{4.259}$$

for any consistent generalized coordinate q; i.e., any $q(\beta, t)$ such that $\Phi(q, \beta, t) = 0$.

■

Theorem 4.7. *Let $A(x)$ be a matrix in which each element $a_{ij}(x) : \mathbb{R}^n \to \mathbb{R}$ is a continuous function. If $rank(A(q)) = n_r$ then there is an open neighborhood $\mathcal{B}(q, \rho)$ of q in which $rank(A(x)) = n_r$ for all $x \in \mathcal{B}(q, \rho)$*

Proof. The proof is done by induction on the minors of a largest non-zero determinant of $A(q)$. Since $rank(A(q)) = n_r$, there are n_r^2 elements $a_{i_1,j_1}(q) \ldots, a_{i_1,j_{n_r}}(q) \ldots$, $a_{i_{n_r},j_1}(q) \ldots, a_{i_{n_r},j_{n_r}}(q)$ such that

$$\Delta(q) = \begin{vmatrix} a_{i_1,j_1}(q) & \ldots & a_{i_1,j_{n_r}}(q) \\ \vdots & & \vdots \\ a_{i_{n_r},j_1}(q) & \ldots & a_{i_{n_r},j_{n_r}}(q) \end{vmatrix} \neq 0$$

The determinant $\Delta(q)$ is a continuous function $\Delta(q) : \mathbb{R}^n \to \mathbb{R}$. In order to prove this, it will be shown by induction that each minor [40] of matrix block

$$A_{\substack{i_1,\ldots,i_{n_r} \\ j_1,\ldots,j_{n_r}}}(x) = \begin{vmatrix} a_{i_1,j_1}(x) & \ldots & a_{i_1,j_{n_r}}(x) \\ \vdots & & \vdots \\ a_{i_{n_r},j_1}(x) & \ldots & a_{i_{n_r},j_{n_r}}(x) \end{vmatrix}$$

consisting of elements situated on rows $i_1, \ldots, i_{n_r}$ and columns $j_1, \ldots, j_{n_r}$, is a continuous function. An order one minor $\Delta_{\substack{k \\ l}}(x)$ of $A_{\substack{i_1,\ldots,i_{n_r} \\ j_1,\ldots,j_{n_r}}}(x)$ is [40] $\Delta_{\substack{k \\ l}}(x) = a_{i_k,j_l}(x)$,

$k, l \in \{1, \ldots, n_r\}$. Since $a_{i,j}(x)$ are continuous, $\Delta_{\substack{k \\ l}}(x)$, $k, l \in \{1, \ldots, n_r\}$, are continuous. Assume, as induction hypothesis, that all minors of order p, $\Delta_{\substack{k_1,\ldots,k_p \\ l_1,\ldots,l_p}}$, are continuous functions. Then, any minor of order $p+1$ has the form [40]

$$\begin{aligned} \Delta_{\substack{k_1,\ldots,k_{p+1} \\ l_1,\ldots,l_{p+1}}}(x) &= a_{k_1,l_1}(x)\Delta_{\substack{k_2,\ldots,k_{p+1} \\ l_2,\ldots,l_{p+1}}}(x) - a_{k_1,l_2}(x)\Delta_{\substack{k_2,\ldots,k_{p+1} \\ l_1,l_3,\ldots,l_{p+1}}}(x) \\ &+ \ldots + (-1)^p a_{k_1,l_{p+1}}(x)\Delta_{\substack{k_2,\ldots,k_{p+1} \\ l_1,\ldots,l_p}}(x) \end{aligned}$$

where $a_{k_1,l_j}(x), j = 1, 2, \ldots, p+1$, are continuous functions, according to the hypothesis, and all minors of order p

$$\Delta_{\substack{k_2,\ldots,k_{p+1} \\ l_2,\ldots,l_{p+1}}}(x), \ldots, \Delta_{\substack{k_2,\ldots,k_{p+1} \\ l_1,\ldots,l_p}}(x)$$

are continuous by the induction hypothesis. Since the product of two continuous functions is a continuous function [44] and the sum of $p+1$ continuous functions is a continuous function [44], terms of the form

$$a_{k_1,l_1}(x)\Delta_{\substack{k_2,\ldots,k_{p+1} \\ l_2,\ldots,l_{p+1}}}(x), -a_{k_1,l_2}(x)\Delta_{\substack{k_2,\ldots,k_{p+1} \\ l_1,l_3,\ldots,l_{p+1}}}(x), \ldots, (-1)^p a_{k_1,l_{p+1}}(x)\Delta_{\substack{k_2,\ldots,k_{p+1} \\ l_1,\ldots,l_p}}(x)$$

are continuous functions. Hence, their sum, $\Delta_{\substack{k_1,\ldots,k_{p+1} \\ l_1,\ldots,l_{p+1}}}$, which is a minor of order $p+1$, is a continuous function. Therefore, all minors of order one are continuous, and the assumption that order p minors are continuous implies that order $p+1$ minors are continuous. By induction, it follows that all minors of order $1, 2, \ldots, n_r$ of matrix block $A_{\substack{i_1,\ldots,i_{n_r} \\ j_1,\ldots,j_{n_r}}}(x)$, are continuous functions. Since the only minor of $A_{\substack{i_1,\ldots,i_{n_r} \\ j_1,\ldots,j_{n_r}}}(x)$ of order n_r is $\Delta(x)$, it follows that $\Delta(x) = det(A_{\substack{i_1,\ldots,i_{n_r} \\ j_1,\ldots,j_{n_r}}}(x))$ is continuous. As a result, by definition of continuity [44], for any $\epsilon > 0$, there is an open neighborhood $\mathcal{B}(q, \rho)$, with $\rho > 0$ depending on q and ϵ, such that $|\Delta(x) - \Delta(q)| < \epsilon$ for any $x \in \mathcal{B}(q, \rho(q, \epsilon))$;

i.e., $\Delta(q) - \epsilon < \Delta(x) < \Delta(q) + \epsilon$, where $\Delta(q)$ is non-zero, by hypothesis. If $\Delta(q) < 0$, let $\epsilon < -\Delta(q)$. Therefore, $\Delta(x) < \Delta(q) + \epsilon < 0$ for any $x \in \mathcal{B}(q, \rho(q, \epsilon))$. If $\Delta(q) > 0$, let $\epsilon < \Delta(q)$. Then, $0 < \Delta(q) - \epsilon < \Delta(x)$, for any $x \in \mathcal{B}(q, \rho(q, \epsilon))$. Hence, $\Delta(x)$ is non-zero for any x in the neighborhood $\mathcal{B}(q, \rho(q, \epsilon))$. As a result, the rank n_r is preserved in the entire neighborhood $\mathcal{B}(q, \rho(q, \epsilon))$. ■

Theorem 4.8. *The rank of the* $2(n+m) \times 2(n+m)$ *matrix*

$$A(q) = \begin{pmatrix} {\Phi_q}^\top & 0 & 0 & I_n \\ 0 & {\Phi_q}^\top & M & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{4.260}$$

is constant and equal to $max(n+m, n+6n_b)$*, for any consistent generalized coordinate* q*; i.e., any* $q(\beta, t)$ *such that* $\Phi(q, \beta, t) = 0$*.*

Proof. By row permutations P_r and column permutations P_c, matrix block

$${\Phi_q}^{I,0} = \begin{pmatrix} I_n & 0 \\ 0 & {\Phi_q}^\top \end{pmatrix} \tag{4.261}$$

can be transformed to

$$P_r {\Phi_q}^{I,0} P_c = \begin{pmatrix} I_n & 0 \\ 0 & {\Phi_u}^\top \\ 0 & {\Phi_v}^\top \end{pmatrix} \tag{4.262}$$

where $m \times m$ matrix Φ_u is non-singular [27]. Consider the $(n+m) \times (n+m)$ matrix block

$${\Phi_u}^{I,0} = \begin{pmatrix} I_n & 0 \\ 0 & {\Phi_u}^\top \end{pmatrix} \tag{4.263}$$

and assume there is a $(n+m)$ dimensional vector

$$v = \begin{pmatrix} v_1 \\ \\ v_2 \end{pmatrix} \tag{4.264}$$

such that

$$\Phi_u{}^{I,0} v = \begin{pmatrix} I_n & 0 \\ \\ 0 & \Phi_u{}^{\top} \end{pmatrix} \begin{pmatrix} v_1 \\ \\ v_2 \end{pmatrix} = \begin{pmatrix} v_1 \\ \\ \Phi_u{}^{\top} v_2 \end{pmatrix} = 0 \tag{4.265}$$

That is,

$$v_1 = 0$$

and

$$\Phi_u{}^{\top} v_2 = 0$$

which implies $v_2 = 0$, since Φ_u is non-singular. Thus, $\Phi_u{}^{I,0}$ is non-singular. Consider matrix blocks

$$M^{\Phi} = \begin{pmatrix} \Phi_q{}^{\top} & 0 \\ \\ 0 & M \end{pmatrix}$$

and

$$\Phi^{q,q} = \begin{pmatrix} \Phi_q{}^{\top} & 0 \\ \\ 0 & \Phi_q{}^{\top} \end{pmatrix}$$

both of which have at most $n+m$ columns. Therefore, their rank cannot exceed $n+m$. Consider the rank of the matrix block

$$\Phi_u{}^{i,q} = \begin{pmatrix} I_n & 0 & \Phi_q{}^{\top} \\ \\ 0 & \Phi_u{}^{\top} & 0 \\ \\ 0 & \Phi_v{}^{\top} & 0 \end{pmatrix}$$

Selecting any number of columns l, equal at most to the number d of degrees of freedom, from block

$$\begin{pmatrix} {\Phi_q}^\top \\ 0 \\ 0 \end{pmatrix}$$

and any l rows from block

$$\begin{pmatrix} 0 & {\Phi_v}^\top & 0 \end{pmatrix}$$

the following matrix block is obtained:

$${\Phi_u}^{I,l} = \begin{pmatrix} I_n & 0 & C_{13} \\ 0 & {\Phi_u}^\top & 0 \\ 0 & C_{32} & 0 \end{pmatrix}$$

where C_{13} is a $n \times l$ matrix obtained by selecting any l columns from matrix ${\Phi_q}^\top$ and C_{32} is a $l \times m$ matrix obtained by selecting any l rows from matrix ${\Phi_v}^\top$. Then, there is a $(n+m+l)$-dimensional vector

$$v = \begin{pmatrix} {v_1}^\top & {v_2}^\top & {v_3}^\top \end{pmatrix}^\top \tag{4.266}$$

such that

$${\Phi_u}^{I,l} v = \begin{pmatrix} I_n & 0 & C_{13} \\ 0 & {\Phi_u}^\top & 0 \\ 0 & C_{32} & 0 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} = 0 \tag{4.267}$$

That is,

$$\begin{aligned} v_1 + C_{13}v_3 &= 0 \\ {\Phi_u}^{\top} v_2 &= 0 \\ C_{32}v_2 &= 0 \end{aligned}$$

Any vector $v = \begin{pmatrix} -(C_{13}v_3)^{\top} & 0 & {v_3}^{\top} \end{pmatrix}^{\top}$ with $v_3 \neq 0$ satisfies the condition of Eq. (4.267). Therefore, ${\Phi_u}^{I,l}$ is singular for any integer $1 \leq l \leq d$, while block ${\Phi_u}^{I,0}$ is non-singular. Hence,

$$rank({\Phi_u}^{I,q}) = n + m$$

The rank of the remaining matrix block

$$M^I = \begin{pmatrix} I_n & 0 \\ 0 & M \end{pmatrix}$$

is $n+6n_b$, as shown in Theorem 4.6. As a result, $rank({\Phi_u}^{I,q}) = n+m$, $rank(M^{\Phi}) \leq n+m$, $rank(\Phi^{q,q}) \leq n+m$, and $rank(M^I) = n + 6n_b$. Consequently,

$$rank(A) = max(n + m, n + 6n_b)$$

■

The rank of matrix A depends only on the topology of the multibody system;i.e., on the number of bodies n_b and number of constraints m. For a system without constraint addition-deletion [28], in which new bodies are not engaged and existent bodies are not disengaged, $rank(A(q))$ is constant at for any $q(\beta, t)$ that satisfies the algebraic constraints $\Phi(q, \beta, t) = 0$.

Definition 4.1. The DAE

$$F(y, y', t) = 0$$

is *uniform index-1* if [13]

1. the index of the constant coefficient system

$$Aw'(t) + Bw(t) = g(t) \tag{4.268}$$

where $A = F_{y'}(\hat{y}, \hat{y}', \hat{t})$ and $B = F_y(\hat{y}, \hat{y}', \hat{t})$, is one for all $(\hat{y}, \hat{y}', \hat{t})$ in a neighborhood of the graph of the solution.

2. A_t, A_y and $A_{y'}$ exist and are bounded in a neighborhood of the solution.

3. $rank(A)$ is constant in a neighborhood of the solution.

The theorem that follows proves that the DAE of the form of Eq. (4.215), for dynamic, direct differentiation, and adjoint equations, are uniform index-1 DAE.

Theorem 4.9. *Assuming M and $\Phi \in C^2(\Omega)$, $\Omega \subseteq \mathbb{R}^n \times \mathbb{R}^{n_\beta} \times \mathbb{R}$, the DAE formulations of Eqs.* (4.20)*,* (4.113)*, and* (4.194)*, for the dynamic, Direct Differentiation, and Adjoint DAE, respectively, are uniform index-1.*

Proof. For index-1 DAE formulations of Eqs. (4.20), (4.113), and (4.194), partial derivatives $F_{y'}$ and F_y, are

$$A = F_{y'} = \begin{pmatrix} {\Phi_q}^\top & 0 & 0 & I \\ 0 & {\Phi_q}^\top & M & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{4.269}$$

and

$$B = F_y = \begin{pmatrix} 0 & 0 & -I & B_{14} \\ 0 & 0 & B_{23} & B_{24} \\ 0 & 0 & \Phi_q & (\Phi_q)' \\ 0 & 0 & 0 & \Phi_q \end{pmatrix} \tag{4.270}$$

Matrix $B_{14} = (\Phi_q{}^\top \chi')_q$ for the index-1 DAE formulation of dynamics of Eq. (4.20), and $B_{14} = 0$ for index-1 DAE formulations of direct differentiation and adjoint equations of Eqs. (4.113) and (4.194). Matrix $B_{23} = -S_v^1$ for the index-1 DAE formulation of dynamics of Eq. (4.20), $B_{23} = -Q^{x_v}$ for the index-1 DAE formulation of direct differentiation equations of Eq. (4.113), and $B_{23} = -Q^{\mu_v}$ for the index-1 DAE formulation of adjoint equations of Eq. (4.194). Matrix $B_{24} = (Mv' + \Phi_q{}^\top \Psi' - S^1)_q$ for the index-1 DAE formulation of dynamic equations of Eq. (4.20), $B_{24} = -Q^{x_q}$ for the index-1 DAE formulation of direct differentiation equations of Eq. (4.113), and $B_{24} = -Q^{\mu_q}$ for the index-1 DAE formulation of adjoint equations of Eq. (4.194). The constant coefficient DAE

$$Aw' + Bw = g$$

is expanded as

$$\Phi_q{}^\top w_1' + w_4' - w_3 + B_{14} w_4 = g_1 \tag{4.271}$$

$$\Phi_q{}^\top w_2' + M w_3' + B_{23} w_3 + B_{24} w_4 = g_2 \tag{4.272}$$

$$\Phi_q w_3 + \Phi_q{}' w_4 = g_3 \tag{4.273}$$

$$\Phi_q w_4 = g_4 \tag{4.274}$$

where $w = \begin{pmatrix} {w_1}^\top & {w_2}^\top & {w_3}^\top & {w_4}^\top \end{pmatrix}^\top$ and $g = \begin{pmatrix} {g_1}^\top & {g_2}^\top & {g_3}^\top & {g_4}^\top \end{pmatrix}^\top$. Differentiating the algebraic constants of Eqs. (4.273) and (4.274) once, the following differential equation is obtained:

$$\begin{pmatrix} {\Phi_q}^\top & 0 & 0 & I \\ 0 & {\Phi_q}^\top & M & 0 \\ 0 & 0 & \Phi_q & (\Phi_q)' \\ 0 & 0 & 0 & \Phi_q \end{pmatrix} \begin{pmatrix} {w_1}' \\ {w_2}' \\ {w_3}' \\ {w_4}' \end{pmatrix} + \begin{pmatrix} 0 & 0 & -I & B_{14} \\ 0 & 0 & B_{23} & B_{24} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{pmatrix} = \begin{pmatrix} g_1 \\ g_2 \\ g_3 \\ g_4 \end{pmatrix} \tag{4.275}$$

where coefficient matrix

$$A_1 = \begin{pmatrix} {\Phi_q}^\top & 0 & 0 & I \\ 0 & {\Phi_q}^\top & M & 0 \\ 0 & 0 & \Phi_q & (\Phi_q)' \\ 0 & 0 & 0 & \Phi_q \end{pmatrix} \tag{4.276}$$

is non-singular. Assuming the contrary, let $v = \begin{pmatrix} {v_1}^\top & {v_2}^\top & {v_3}^\top & {v_4}^\top \end{pmatrix}^\top$ be a vector belonging to the null-space of matrix A_1; i.e.,

$$A_1 v = \begin{pmatrix} {\Phi_q}^\top & 0 & 0 & I \\ 0 & {\Phi_q}^\top & M & 0 \\ 0 & 0 & \Phi_q & (\Phi_q)' \\ 0 & 0 & 0 & \Phi_q \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix} = 0 \tag{4.277}$$

Therefore,

$$\Phi_q{}^{\top} v_1 + v_4 = 0 \tag{4.278}$$

$$\Phi_q{}^{\top} v_2 + M v_3 = 0 \tag{4.279}$$

$$\Phi_q v_3 + \Phi_q{}' v_4 = 0 \tag{4.280}$$

$$\Phi_q v_4 = 0 \tag{4.281}$$

Component v_4 must be zero. Otherwise, pre-multiplying Eq. (4.278) by $v_4{}^{\top}$

$$v_4{}^{\top} \Phi_q{}^{\top} + \|v_4\|_2^2 = 0 \tag{4.282}$$

and accounting for the identity of Eq. (4.281), $\Phi_q v_4 = 0$, it follows that

$$\|v_4\|_2^2 = 0$$

which implies that $v_4 = 0$. As a result, Eq. (4.278) is reduced to

$$\Phi_q{}^{\top} v_1 = 0$$

which implies that $v_1 = 0$, since $\Phi_q{}^{\top}$ has full column rank. The remaining two equations, Eqs. (4.279) and (4.280), are reduced to

$$\begin{pmatrix} M & \Phi_q{}^{\top} \\ \Phi_q & 0 \end{pmatrix} \begin{pmatrix} v_3 \\ v_2 \end{pmatrix} = 0$$

Since the Schur complement matrix

$$M^S = \begin{pmatrix} M & \Phi_q{}^{\top} \\ \Phi_q & 0 \end{pmatrix}$$

is non-singular [52], it follows that $v_3 = 0$ and $v_2 = 0$. Consequently $v = 0$, which implies that A_1 is non-singular. Hence, the differential equation of Eq. (4.275), obtained by once differentiating the constraints of Eqs. (4.273) and (4.274) of the DAE of Eqs. (4.271) through (4.274), is an ODE. Therefore, the DAE of Eqs. (4.271) through (4.274) is an index-1 DAE [13]. Existence and boundedness of A_t, A_y, and $A_{y'}$ follow from the hypothesis $M(q, \beta, t)$ and $\Phi(q, \beta, t) \in C^2(\Omega), \Omega \subseteq \mathbb{R}^n \times \mathbb{R}^{n_\beta} \times \mathbb{R}$. As a result of Theorem 4.8,

$$rank(A(q)) = max(n + m, n + 6n_b)$$

and Theorem 4.7 shows that a matrix function of q, with continuous elements, preserves its rank in a neighborhood $\mathcal{B}(q, \rho)$. Therefore, $rank(A(x))$ is constant in a neighborhood $\mathcal{B}(q, \rho)$ of the graph of the solution. Properties 1 through 3 of Definition 4.1 are, therefore, satisfied. Thus, the DAE formulations of Eqs. (4.20), (4.113), and (4.194) for dynamic, direct differentiation, and adjoint DAE, respectively, are uniform index-1. ■

According to Theorem 3.2.1. of Ref. [13], a BDF method of order up to six is convergent for a uniform index-1 DAE. Hence, an up to order six BDF integration method applied to Eqs. (4.20), (4.113), and (4.194), which were shown to be uniform index-1, has a unique solution.

CHAPTER 5
COARSE GRAINED PARALLELISM: PIECEWISE SOLUTION OF THE ADJOINT DIFFERENTIAL-ALGEBRAIC EQUATIONS

Consider a functional subjected to optimization or optimal control, of the form

$$\Psi(\beta, t^2) = l(t^2, q^2, {q'}^2, \beta) + \int_{t^1}^{t^2} g(q, q', \lambda, \beta, t)dt \tag{5.1}$$

with final time t^2 implicitly defined by the condition [26]

$$\Omega(t^2, q(\beta, t^2), \beta) = 0 \tag{5.2}$$

The index-3 adjoint DAE of a multibody system with respect to the functional of Eq. (5.1) is [26]

$$M\mu'' + D_1\mu' + D_2\mu + {\Phi_q}^\top \nu = s_a \tag{5.3}$$

$$\Phi_q \mu = r_a \tag{5.4}$$

where μ is the adjoint variable, ν is the adjoint Lagrange multiplier, and

$$D_2 = \left[M'' + \left(\frac{d}{dt}(S_v^1)\right)^\top - (S_q^1)^\top + \left(({\Phi_q}^\top \lambda)_q{}^\top\right) + \left((Mv')_q\right)^\top\right] \tag{5.5}$$

$$D_1 = (2M' + {S_v^1}^\top) \tag{5.6}$$

$$s_a = {g_q}^\top - \frac{d}{dt}({g_{q'}}^\top) \tag{5.7}$$

$$r_a = {g_\lambda}^\top \tag{5.8}$$

The constraint Jacobian Φ_q is assumed to have full row-rank at all times [27]. As a result [58], there exist permutation matrices P_r and P_c [18]; i.e., orthogonal matrices

with columns having all elements zero except one element, which is unity, such that

$$P_r \Phi_q P_c = \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} \tag{5.9}$$

where Φ_u is non-singular.

Definition 5.1. Define the *partitioned adjoint variable*

$$\mu^\pi \equiv {P_c}^\top \mu \tag{5.10}$$

the *dependent adjoint variable*

$$\mu^u \equiv \begin{pmatrix} I_m & 0 \end{pmatrix} \mu^\pi \tag{5.11}$$

and the *independent adjoint variable*

$$\mu^v \equiv \begin{pmatrix} 0 & I_d \end{pmatrix} \mu^\pi \tag{5.12}$$

where I_m is the $m \times m$ identity matrix and I_d is the $d \times d$ identity matrix.

In matrix form,

$$\begin{pmatrix} \mu^u \\ \mu^v \end{pmatrix} = \begin{pmatrix} I_m & 0 \\ 0 & I_d \end{pmatrix} \mu^\pi = \mu^\pi \tag{5.13}$$

and, since P_c is an orthogonal matrix; i.e., $(P_c)^{-1} = {P_c}^\top$,

$$\mu = P_c \mu^\pi = P_c \begin{pmatrix} \mu^u \\ \mu^v \end{pmatrix} \tag{5.14}$$

Substituting for μ^π defined by Eq. (5.10) into Eq. (5.12), the independent adjoint variable μ^v is expressed as a function of the adjoint variable μ as

$$\mu^v = \begin{pmatrix} 0 & I_d \end{pmatrix} {P_c}^\top \mu \tag{5.15}$$

Differentiating Eq. (5.15), $\mu^{v\prime}$ is expressed as a function of the derivative μ' of the adjoint variable,

$$\mu^{v\prime} = \begin{pmatrix} 0 & I_d \end{pmatrix} {P_c}^{\top} \mu' \tag{5.16}$$

5.1 The Effect of Row and Column Permutations due to the Constraint Jacobian Factorization on the Adjoint Underlying ODE

According to Theorem 3.5 in Chapter 3, when permutation matrices P_r and P_c are $m \times m$ and $n \times n$ identity matrices, respectively, the adjoint coordinate partitioning underlying ODE (CPUODE) of Eq. (3.105) is obtained by pre-multiplying Eq. (5.3) by ${X_0}^{\top}$, where matrix X_0 is defined by Eq. (A.50) in the Appendix, and performing the change of variable $\mu^v = \begin{pmatrix} 0 & I_d \end{pmatrix} \mu$. In general, however, partitions of the constraint Jacobian Φ_q of the form $\begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix}$, with Φ_u non-singular, require [58] row and colum permutations; i.e., matrices P_r and P_c of Eq. (5.9) are different from the $m \times m$ and $n \times n$ identity matrices, respectively. In order to obtain the CPUODE when the constraint Jacobian has the form of Eq. (5.9), where permutation matrices P_r and P_c are constant during the foregoing partitioning [58], the theorems that follow are proven.

Theorem 5.1. *Matrix ${X_0}^{\top} {P_c}^{\top} M P_c X_0$, where M is the mass matrix and X_0 is defined by Eq.* (A.50) *in the Appendix, is non-singular.*

Proof. Define

$$W_0 = P_c X_0 \tag{5.17}$$

Since, according to the identity of Eq. (5.9),

$$\Phi_q P_c = {P_r}^{-1} \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix}$$

the product identity

$$\Phi_q W_0 = \Phi_q P_c X_0 = {P_r}^{-1} \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} \begin{pmatrix} -{\Phi_u}^{-1}\Phi_v \\ I_d \end{pmatrix} = 0 \tag{5.18}$$

follows. Also, the product

$${W_0}^{\top} W_0 = {X_0}^{\top} {P_c}^{\top} P_c X_0 = {X_0}^{\top} X_0 \tag{5.19}$$

is non-singular, since ${X_0}^{\top} X_0$ is positive definite, as shown by Corollary 3.4 in Chapter 3. Therefore, ${W_0}^{\top}$ and W_0 have the properties of T_1 and T_2, respectively, defined by Theorem 3.1 of Chapter 3. Hence, according to Theorem 3.1 3, the product ${W_0}^{\top} M W_0$ is non-singular; i.e., ${X_0}^{\top} {P_c}^{\top} M P_c X_0$ is non-singular. ■

Theorem 5.2. *The equation obtained by pre-multiplying Eq.* (5.3) *by* ${W_0}^{\top}$*, where matrix* W_0 *is defined by Eq.* (5.17) *and performing the change of variable of Eq.* (5.15)*, is an ODE.*

Proof. Pre-multiplying the adjoint differential equation of Eq. (5.3) by ${W_0}^{\top}$ and accounting for the property that

$${W_0}^{\top} {\Phi_q}^{\top} = 0$$

which is a result of transposing the identity of Eq. (5.18), the term containing the adjoint Lagrange multiplier ν vanishes. Hence, Eq. (5.3) is re-written as

$$W_0{}^\top M\mu'' + W_0{}^\top D_1\mu' - W_0{}^\top D_2\mu = W_0{}^\top s_a \tag{5.20}$$

Pre-multiplying Eq. (5.9) by $P_r{}^\top$, post-multiplying it by $P_c{}^\top$, and accounting for orthogonality of permutation matrices P_r and P_c; i.e., $P_r{}^{-1} = P_r{}^\top and P_c{}^{-1} = P_c{}^\top$,

$$\Phi_q = P_r{}^\top \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} P_c{}^\top \tag{5.21}$$

Substituting for Φ_q from Eq. (5.21) into Eq. (5.4),

$$P_r{}^\top \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} P_c{}^\top \mu - r_a = 0 \tag{5.22}$$

Pre-multiplying Eq. (5.22) by orthogonal matrix P_r and using the definition of Eq. (5.10) and the identity of Eq. (5.13), $P_c{}^\top \mu = \mu^\pi = \begin{pmatrix} \mu^u \\ \mu^v \end{pmatrix}$,

$$\begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} \begin{pmatrix} \mu^u \\ \mu^v \end{pmatrix} = P_r r_a \tag{5.23}$$

Expanding terms,

$$\Phi_u\mu^u + \Phi_v\mu^v = P_r r_a$$

and pre-multiplying by non-singular matrix Φ_u, the dependent adjoint variable μ^u is expressed as a linear function of the independent adjoint variable μ^v as

$$\mu^u = -\Phi_u{}^{-1}\Phi_v\mu^v + \Phi_u{}^{-1}P_r r_a \tag{5.24}$$

Substituting this result for μ^u into the identity of Eq. (5.14), the adjoint variable μ is expressed as a linear function of the independent adjoint variable μ^v as

$$\begin{aligned} \mu &= P_c \begin{pmatrix} \mu^u \\ \\ \mu^v \end{pmatrix} = P_c \begin{pmatrix} -\Phi_u^{-1}\Phi_v\mu^v \\ \\ \mu^v \end{pmatrix} + P_c \begin{pmatrix} \Phi_u^{-1}P_r r_a \\ \\ 0 \end{pmatrix} \\ &= P_c X_0 \mu^v + P_c X_1 P_r r_a \end{aligned} \tag{5.25}$$

where matrix X_1 is defined by Eq. (A.51) in the Appendix. Differentiating Eq. (5.25),

$$\mu' = P_c X_0 \mu^{v\prime} + P_c X_0{}' \mu^v + P_c (X_1 P_r r_a)' \tag{5.26}$$

and replacing the first derivative of X_0 by Eq. (A.55) in the Appendix, the derivative of the adjoint variable is expressed as a linear function of the independent adjoint variable μ^v and its derivative $\mu^{v\prime}$ as

$$\mu' = P_c X_0 \mu^{v\prime} + P_c X_2 X_0 \mu^v + P_c (X_1 P_r r_a)' \tag{5.27}$$

where X_2 is defined by Eq. (A.56) in the Appendix. Differentiating Eq. (5.27),

$$\mu'' = P_c X_0 \mu^{v\prime\prime} + 2P_c X_0{}' \mu^{v\prime} + P_c X_0{}'' \mu^v + P_c (X_1 P_r r_a)'' \tag{5.28}$$

and substituting for the first and second derivatives of X_0, defined by Eqs. (A.55) and (A.60) in the Appendix, the derivative of the adjoint variable is expressed as a linear function of the independent adjoint variable μ^v and its first and second derivatives $\mu^{v\prime}$ and $\mu^{v\prime\prime}$,

$$\mu'' = P_c X_0 \mu^{v\prime\prime} + 2P_c X_2 X_0 \mu^{v\prime} + P_c X_3 X_0 \mu^v + P_c (X_1 P_r r_a)'' \tag{5.29}$$

where X_3 is defined by Eq. (A.61) in the Appendix.

Substituting for the adjoint variable μ and its derivatives μ' and μ'' from Eqs. (5.25), (5.27), and (5.29) and for matrix W_0 of Eq. (5.17) into Eq. (5.20), the following differential equation is obtained:

$$\begin{aligned} X_0{}^\top P_c{}^\top M P_c X_0 \mu^{v\prime\prime} &+ (2X_0{}^\top P_c{}^\top M P_c X_2 X_0 + X_0{}^\top P_c{}^\top D_1 P_c X_0)\mu^{v\prime} \\ &+ (X_0{}^\top P_c{}^\top M P_c X_3 X_0 + X_0{}^\top P_c{}^\top D_1 P_c X_2 X_0 \\ &+ X_0{}^\top P_c{}^\top D_2 P_c X_0)\mu^v = X_0{}^\top P_c{}^\top s_a \qquad (5.30) \\ &- X_0{}^\top P_c{}^\top M P_c (X_1 P_r r_a)'' - X_0{}^\top P_c{}^\top D_1 P_c (X_1 P_r r_a)' \\ &- X_0{}^\top P_c{}^\top D_2 P_c X_1 P_r r_a \end{aligned}$$

The highest derivative term $\mu^{v\prime\prime}$ is pre-multiplied by matrix $X_0{}^\top P_c{}^\top M P_c X_0$, which, according to Theorem 5.1, is non-singular. Therefore, Eq. (5.30) is an ODE. Since the differential variable μ^v is the independent part of μ^π corresponding to the underlying constraint Jacobian partitioning $P_r \Phi_q P_c = \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix}$, Eq. (5.30) is the CPUODE of the adjoint DAE of Eqs. (5.3) and (5.4). ■

5.2 Linearly Independent Solutions of the Adjoint CPUODE

Consider a sub-interval $\mathcal{I}^j \equiv [t_j, t_{j+1}]$ (the *fine grid*) of the time interval $[t^1, t^2]$ (the *coarse grid*) during which sensitivity analysis of the multibody system is performed. Let $y^j(t) = \begin{pmatrix} y^{j,1}(t) \\ y^{j,2}(t) \end{pmatrix}$ be the solution of the first order form of the

CPUODE of Eq. (5.30),

$$\begin{aligned} y^{j,1\prime} - y^{j,2} &= 0 \\ U_2 y^{j,2\prime} + U_1 y^{j,2} + U_0 y^{j,1} &= u \end{aligned} \tag{5.31}$$

where

$$y^{j,1}(t) \equiv \mu^v(t) \tag{5.32}$$

$$y^{j,2}(t) \equiv \mu^{v\prime}(t) \tag{5.33}$$

and

$$\begin{aligned} u &= X_0{}^\top P_c{}^\top M P_c (X_1 P_r r_a)'' \\ &- X_0{}^\top P_c{}^\top D_1 P_c (X_1 P_r r_a)' - X_0{}^\top P_c{}^\top D_2 P_c X_1 P_r r_a \end{aligned} \tag{5.34}$$

$$U_0 = X_0{}^\top P_c{}^\top M P_c X_3 X_0 + X_0{}^\top P_c{}^\top D_1 P_c X_2 X_0 + X_0{}^\top P_c{}^\top D_2 P_c X_0 \tag{5.35}$$

$$U_1 = 2 X_0{}^\top P_c{}^\top M P_c X_2 X_0 + X_0{}^\top P_c{}^\top D_1 P_c X_0 \tag{5.36}$$

$$U_2 = X_0{}^\top P_c{}^\top M P_c X_0 \tag{5.37}$$

As shown in Chapter 3, the adjoint CPUODE is stable in the backward direction. Therefore, on the fine-grid $\mathcal{I}^j$, initial conditions are specified at time t_{j+1} and integration progresses from t_{j+1} to $t_j < t_{j+1}$. Let $y_i^j(t) = \begin{pmatrix} y_{1,i}^j(t) \\ y_{2,i}^j(t) \end{pmatrix}$ be the solution of the homogeneous IVP of Eq. (5.31),

$$\begin{aligned} y_i^{j,1\prime} - y_i^{j,2} &= 0 \\ U_2 y_i^{j,2\prime} + U_1 y_i^{j,2} + U_0 y_i^{j,1} &= 0 \end{aligned} \tag{5.38}$$

with initial conditions $y_i^j(t_{j+1}) = e_i$, where $2d \times 1$ unit vector e_i has all elements zero except the i-th, which is one. In addition, let $v^j(t) = \begin{pmatrix} \mu^v(t) \\ \\ \mu^{v\prime}(t) \end{pmatrix} = \begin{pmatrix} v_1^j(t) \\ \\ v_2^j(t) \end{pmatrix}$ be a particular solution of the non-homogeneous CPUODE of Eq. (5.31) with zero initial conditions; i.e., $v^j(t_{j+1}) = 0$. Since u, U_0, U_1, and U_2 of Eqs. (5.34) through (5.37) do not depend on the independent adjoint variable μ^v, the CPUODE of Eq. (5.31) is linear. As a result, the general solution of the adjoint CPUODE of Eq. (5.31), on interval $\mathcal{I}^j$, is [4]

$$y^j(t) = Y_0^j(t)s_j + v^j(t) \tag{5.39}$$

where

$$Y_0^j(t) = \begin{pmatrix} y_1^j(t) & \dots & y_{2d}^j(t) \end{pmatrix} \tag{5.40}$$

is the fundamental matrix [4] in which the i-th column, $i = 1, 2, \dots, 2d$, is the vector $y_i^j(t)$; i.e., the solution of the homogeneous IVP of Eq. (5.31) with initial conditions $y_i^j(t_{j+1}) = e_i$. Since the fundamental matrix at time t_{j+1} is the $2d \times 2d$ identity matrix,

$$Y_0^j(t_{j+1}) = \begin{pmatrix} y_1^j(t_{j+1}) & \dots & y_{2d}^j(t_{j+1}) \end{pmatrix} = I_{2d} \tag{5.41}$$

hence, $Y_0^j(t)$ is non-singular for all time [18], $s_j \in \mathbb{R}^{2d}$ is a constant vector to be determined by the initial conditions of the IVP of Eq. (5.31), and $v^j(t)$ is the particular solution of the non-homogeneous CPUODE of Eq. (5.31) with zero initial conditions.

Since the linear ODE of Eq. (5.39) is the adjoint CPUODE of Eq. (5.30) in

first order form, given initial conditions $y^j(t_{j+1}) = y^j_{j+1}$; i.e.,

$$\begin{aligned} y^{j,1}(t_{j+1}) &\equiv \mu^v(t_{j+1}) = y^{j,1}_{j+1} \\ y^{j,2}(t_{j+1}) &\equiv \mu^{v\prime}(t_{j+1}) = y^{j,2}_{j+1} \end{aligned} \tag{5.42}$$

where $y^j_{j+1} = \left(\begin{array}{cc} y^{j,1}_{j+1}{}^\top & y^{j,2}_{j+1}{}^\top \end{array} \right)^\top$, the solution of Eq. (5.39) is obtained through the following algorithm (the *integrate-recover-assemble* algorithm):

1. Integrate the adjoint DAE of Eqs. (5.3) and (5.4) at time $t \in (t_j, t_{j+1})$, in the backward direction, with initial conditions at t_{j+1} obtained by substituting the independent adjoint variable $\mu^v(t_{j+1}) = y^{j,1}_{j+1}$ and its derivative $\mu^{v\prime}(t_{j+1}) = y^{j,2}_{j+1}$ into Eqs. (5.25) and (5.27), respectively,

$$\mu(t_{j+1}) = P_c X_0 y^{j,1}_{j+1} + P_c X_1 P_r r_a \tag{5.43}$$

$$\mu'(t_{j+1}) = P_c X_0 y^{j,2}_{j+1} + P_c X_2 X_0 y^{j,1}_{j+1} + P_c (X_1 P_r r_a)' \tag{5.44}$$

2. Recover the independent adjoint variable $\mu^v(t)$ and its derivative $\mu^{v\prime}(t)$, at time t, by substituting the adjoint variable $\mu(t)$ and its derivative $\mu'(t)$, previously solved for, into Eqs. (5.15) and (5.16), respectively.

3. Assemble the solution of the ODE of Eq. (5.39),

$$y^j(t) = \left(\begin{array}{c} y^{j,1}(t) \\ \\ y^{j,2}(t) \end{array} \right) \tag{5.45}$$

where

$$\begin{aligned} y^{j,1}(t) &= \mu^v(t) \\ y^{j,2}(t) &= \mu^{v\prime}(t) \end{aligned}$$

Alternatively, the solution of the ODE of Eq. (5.39) can be obtained by assembling terms u, U_0, U_1, and U_2 of Eqs. (5.34) through (5.37) and integrating the ODE of Eq. (5.31). The integrate-recover-assemble algorithm has the advantage of eliminating the need for computing u, U_0, U_1, and U_2, which is computationally expensive and using, instead, the adjoint index-1 formulation developed in Section 4.4 of Chapter 4, which involves fewer matrix evaluations. However, integration of an index-1 DAE with solution $y(t)$ of dimension $2(n+m)$ is more expensive than integration of an ODE with solution $y^j(t)$ of dimension $2d = 2(n-m)$.

The integrate-recover-assemble algorithm requires that the adjoint DAE initial conditions defined by Eqs. (5.43) and (5.44) are consistent; i.e., they must satisfy the adjoint position constraint equation of Eq. (5.4) and the adjoint velocity constraint equation of Eq. (4.189).

Theorem 5.3. *Given arbitrary d-dimensional vectors a and b, the n-dimensional vectors*

$$\mu_1 = P_c X_0 a + P_c X_1 P_r r_a \tag{5.46}$$

$$\mu_2 = P_c X_0 b + P_c X_2 X_0 a + P_c (X_1 P_r r_a)' \tag{5.47}$$

satisfy the following equations:

$$\Phi_q \mu_1 - r_a = 0 \tag{5.48}$$

$$\Phi_q \mu_2 + (\Phi_q)' \mu_1 - r_a{}' = 0 \tag{5.49}$$

Proof. Substituting for μ_1 of Eq. (5.46) into the left-side of Eq. (5.48),

$$\Phi_q (P_c X_0 a + P_c X_1 P_r r_a) - r_a = \Phi_q P_c X_0 a + \Phi_q P_c X_1 P_r r_a - r_a \tag{5.50}$$

and accounting for the identity

$$\Phi_q P_c = {P_r}^{\top} \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} \tag{5.51}$$

which is obtained by pre-multiplying Eq. (5.9) by ${P_r}^{-1} = {P_r}^{\top}$, the left-side of Eq. (5.48) is re-written as

$$\begin{aligned} \Phi_q P_c X_0 a + \Phi_q P_c X_1 P_r r_a - r_a &= {P_r}^{\top} \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} X_0 a \\ &+ {P_r}^{\top} \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} X_1 P_r r_a - r_a \end{aligned} \tag{5.52}$$

Substituting for the product $\Phi_q P_c$ defined by Eq. (5.51) and for the matrix X_0 defined by Eq. (A.50) in the Appendix into the product $\Phi_q P_c X_0$,

$$\Phi_q P_c X_0 = \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} X_0 = \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} \begin{pmatrix} -{\Phi_u}^{-1} \Phi_v \\ \\ I_d \end{pmatrix} = 0 \tag{5.53}$$

and substituting for the product $\Phi_q P_c$ defined by Eq. (5.51) and for matrix X_1, defined by Eq. (A.51) in the Appendix into the product $\Phi_q P_c X_1$,

$$\Phi_q P_c X_1 = \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} X_1 = \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} \begin{pmatrix} {\Phi_u}^{-1} \\ \\ 0 \end{pmatrix} = I_m \tag{5.54}$$

where I_m is the $m \times m$ identity matrix. Substituting products of Eqs. (5.53) and (5.54) into Eq. (5.52),

$${P_r}^{\top} \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} X_0 a + {P_r}^{\top} \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} X_1 P_r r_a - r_a = {P_r}^{\top} P_r r_a - r_a = 0 \tag{5.55}$$

since ${P_r}^{\top} P_r = I_m$. Therefore, the identity of Eq. (5.48) is satisfied.

The term $(X_1 P_r r_a)'$ of Eq. (5.47) is expanded as

$$(X_1 P_r r_a)' = {X_1}' P_r r_a + X_1 P_r {r_a}' \tag{5.56}$$

which, accounting for the expression of ${X_1}'$ defined by Eq. (A.66) and for the expression of X_2 defined by Eq. (A.57) in the Appendix, is re-written as

$$(X_1 P_r r_a)' = -X_1 P_r (\Phi_q)' P_c X_1 P_r r_a + X_1 P_r {r_a}' \tag{5.57}$$

Substituting for $(X_1 P_r r_a)'$ of Eq. (5.57) and for X_2 defined by Eq. (A.57) in the Appendix into Eq. (5.47), μ_2 is re-written as

$$\begin{aligned} \mu_2 &= P_c X_0 b - P_c X_1 P_r (\Phi_q)' P_c X_0 a \\ &- P_c X_1 P_r (\Phi_q)' P_c X_1 P_r r_a + P_c X_1 P_r {r_a}' \end{aligned} \tag{5.58}$$

Substituting for μ_1 of Eq. (5.46) and for μ_2 of Eq. (5.58) into the left-side of Eq. (5.49),

$$\begin{aligned} \Phi_q \mu_2 + (\Phi_q)' \mu_1 - {r_a}' &= \Phi_q (P_c X_0 b - P_c X_1 P_r (\Phi_q)' P_c X_0 a \\ &- P_c X_1 P_r (\Phi_q)' P_c X_1 P_r r_a + P_c X_1 P_r {r_a}') \\ &+ (\Phi_q)' (P_c X_0 a + P_c X_1 P_r r_a) - {r_a}' \end{aligned} \tag{5.59}$$

and grouping terms,

$$\begin{aligned}
\Phi_q\mu_2 + (\Phi_q)'\mu_1 - r_a{}' &= \Phi_q P_c X_0 b \\
&+ ((\Phi_q)' P_c X_0 - \Phi_q P_c X_1 P_r (\Phi_q)' P_c X_0) a \\
&+ ((\Phi_q)' P_c X_1 P_r - \Phi_q P_c X_1 P_r (\Phi_q)' P_c X_1 P_r) r_a \\
&- (I_m - \Phi_q P_c X_1 P_r) r_a{}' \\
&= \Phi_q P_c X_0 b \\
&+ (I_m - \Phi_q P_c X_1 P_r)(\Phi_q)' P_c X_0 a \\
&+ (I_m - \Phi_q P_c X_1 P_r)(\Phi_q)' P_c X_1 P_r r_a \\
&- (I_m - \Phi_q P_c X_1 P_r) r_a{}' \qquad (5.60)
\end{aligned}$$

where I_m is the $m \times m$ identity matrix. Therefore,

$$\begin{aligned}
\Phi_q\mu_2 &+ (\Phi_q)'\mu_1 - r_a{}' = \Phi_q P_c X_0 b \\
&+ (I_m - \Phi_q P_c X_1 P_r)((\Phi_q)' P_c X_0 a + (\Phi_q)' P_c X_1 P_r r_a - r_a{}') \qquad (5.61)
\end{aligned}$$

Substituting for the product $\Phi_q P_c$ defined by Eq. (5.51) into factor $I_m - \Phi_q P_c X_1 P_r$,

$$I_m - \Phi_q P_c X_1 P_r = I_m - P_r{}^\top \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} X_1 P_r \qquad (5.62)$$

and substituting for the product $\begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} X_1$ defined by Eq. (5.54) into Eq. (5.62),

$$I_m - \Phi_q P_c X_1 P_r = I_m - P_r{}^\top I_m P_r = 0 \qquad (5.63)$$

since $P_r{}^\top P_r = P_r{}^{-1} P_r = I_m$. Substituting for the identity of Eq. (5.53) and for the identity of Eq. (5.63) into Eq. (5.61), the left-side of Eq. (5.49) is identically zero.

As a result, Eq. (5.49) is satisfied. ■

Corollary 5.4. *Given arbitrary d-dimensional vectors a and b, the n-dimensional vectors*

$$\mu_1 = P_c X_0 a \tag{5.64}$$

$$\mu_2 = P_c X_0 b + P_c X_2 X_0 a \tag{5.65}$$

satisfy

$$\Phi_q \mu_1 = 0 \tag{5.66}$$

$$\Phi_q \mu_2 + (\Phi_q)' \mu_1 = 0 \tag{5.67}$$

Proof. Substituting for r_a and $r_a{}'$ the m-dimensional zero vector into Eqs. (5.46) and (5.47), it follows from Theorem 5.3 that Eqs. (5.48) and (5.49), in which r_a and $r_a{}'$ are replaced by zero, are satisfied; i.e. Eqs. (5.66) and (5.67) are satisfied. ■

The linearly independent columns $y_i^j(t)$, $i = 1, 2, \ldots, 2d$, of the fundamental matrix $Y_0^j(t)$ are obtained by applying the integrate-recover-assemble algorithm to the homogeneous adjoint DAE; i.e., the DAE obtained by substituting the right sides s_a and r_a of Eqs. (5.3) and (5.4) by zero. The initial conditions for the homogeneous adjoint DAE have the form of Eqs. (5.43) and (5.44), in which r_a is replaced by zero and $\begin{pmatrix} {y_{j+1}^{j,1}}^\top & {y_{j+1}^{j,2}}^\top \end{pmatrix}^\top = e_i$; i.e., $y_{j+1}^{j,1} = e_i^1$ and $y_{j+1}^{j,2} = e_i^2$, where e_i^1 is the vector consisting of the first d elements of unit vector e_i and e_i^2 is the vector consisting of

the last d elements of unit vector e_i,

$$\mu(t_{j+1}) = P_c X_0 e_i^1 \tag{5.68}$$

$$\mu'(t_{j+1}) = P_c X_0 e_i^2 + P_c X_2 X_0 e_i^1 \tag{5.69}$$

According to Corollary 5.4, initial conditions of Eqs. (5.68) and (5.69) are consistent. After recovering the independent adjoint variable $\mu^v(t)$ and its derivative $\mu^{v\prime}(t)$ from the adjoint variable $\mu(t)$ and its derivative $\mu'(t)$, the i-th column $y_i^j(t)$ of the fundamental matrix $Y_0^j(t)$ is assembled at time t. Each such column is, therefore, computed independently. The particular solution $v^j(t)$ is independently obtained by applying the integrate-recover-assemble algorithm to the non-homogeneous adjoint DAE of Eqs. (5.3) and (5.4), with initial conditions of the form of Eqs. (5.43) and (5.44), in which $y_{j+1}^{j,1} = 0$ and $y_{j+1}^{j,2} = 0$; i.e.,

$$\mu(t_{j+1}) = P_c X_1 P_r r_a \tag{5.70}$$

$$\mu'(t_{j+1}) = P_c (X_1 P_r r_a)' \tag{5.71}$$

According to Theorem 5.3, initial conditions of Eqs. (5.70) and (5.71) are consistent. Application of the integrate-recover-assemble algorithm for independently evaluating columns $y_i^j(t)$, $i = 1, 2, \ldots, 2d$, of the fundamental matrix $Y_0^j(t)$, and using the particular solution $v^j(t)$, represents a parallel algorithm with $2d$ independent threads for computing the columns of $Y_0^j(t)$ and one independent thread for computing the particular solution $v^j(t)$. The $2d+1$ parallel thread algorithm thus obtained is coarse-grained, according to the convention adopted in Chapter 2. It should be noted that evaluation at time $t \in [t_j, t_{j+1}]$ of matrix terms involved in the adjoint DAE index-

1 formulation developed in Section 4.4 of Chapter 4 requires the evaluation of the generalized coordinate vector $q(t)$, its derivatives $q'(t)$ and $q''(t)$, and the Lagrange multiplier $\lambda(t)$. This is achieved by integrating in advance the index-1 DAE formulation of the equations of motion developed in Section 4.2 of Chapter 4; storing $q(t_k)$, $q'(t_k)$, $q''(t_k)$, and $\lambda(t_k)$ computed on a mesh

$$\Delta^j \equiv \{t_{k_1} = t_j, \ldots, t_{k_2} = t_{j+1}\} \tag{5.72}$$

that covers interval $\mathcal{I}^j = [t_j, t_{j+1}]$; and interpolating for q, q', q'', and λ at the required time $t \in [t_j, t_{j+1}]$ using the stored $q(t_k)$, $q'(t_k)$, $q''(t_k)$, and $\lambda(t_k)$, for each $t_k \in \Delta^j$.

Therefore, on each fine-grid sub-interval $\mathcal{I}^j = [t_j, t_{j+1}]$, $j = 1, 2, \ldots, N_j$, where $t_1 \equiv t^1$ and $t_{N_j+1} \equiv t^2$, $2d + 1$ independent IVP must be solved. Vectors s_j are determined [4] so that the adjoint CPUODE numerical solution $z(t) \equiv y^j(t), t \in \mathcal{I}^j$, where $y^j(t)$ is the *piecewise* adjoint CPUODE solution defined by Eq. (5.39), is continuous over the entire interval $[t^1, t^2]$; i.e.,

$$y^j(t_{j+1}) = y^{j+1}(t_{j+1}), j = 1, 2, \ldots, N_j - 1 \tag{5.73}$$

Substituting for $y^j(t)$ by its expression of Eq. (5.39),

$$Y_0^j(t_{j+1})s_j + v^j(t_{j+1}) = Y_0^{j+1}(t_{j+1})s_{j+1} + v^{j+1}(t_{j+1}), j = 1, 2, \ldots, N_j - 1 \tag{5.74}$$

and ordering terms,

$$-Y_0^{j+1}(t_{j+1})s_{j+1} + s_j = v^{j+1}(t_{j+1}), j = 1, 2, \ldots, N_j - 1 \tag{5.75}$$

At final time $t = t^2 \equiv t_{N_j+1}$,

$$y(t_{N_j+1}) = Y_0^{N_j}(t_{N_j+1})s_{N_j} + v^{N_j}(t_{N_j+1}) = y^2 \tag{5.76}$$

where y^2 is the state vector y^{N_j} of the last fine-grid $\mathcal{I}^{N_j}$ at final time $t_{N_j+1} \equiv t^2$. Vector y^2 is calculated using the adjoint variable and its derivative at final time, as follows:

1. Compute $\mu(t^2)$ and $\mu'(t^2)$ by solving [26], at final time t^2, linear systems of Eqs. (4.197) and (4.198).

2. Recover the independent adjoint variable $\mu^v(t^2)$ and its derivative $\mu^{v\prime}(t^2)$ at final time t^2 by substituting for the adjoint variable $\mu(t^2)$ and its derivative $\mu'(t^2)$ into Eqs. (5.15) and (5.16), respectively.

3. Assemble the solution of the ODE of Eq. (5.39) at final time t^2,

$$y^j(t^2) = \begin{pmatrix} y^{j,1}(t^2) \\ \\ y^{j,2}(t^2) \end{pmatrix} \tag{5.77}$$

where

$$\begin{aligned} y^{j,1}(t^2) &= \mu^v(t^2) \\ y^{j,2}(t^2) &= \mu^{v\prime}(t^2) \end{aligned}$$

As a result, Eqs. (5.75) and (5.76) form the linear system.

$$\begin{pmatrix} I & -Y_0^2(t_2) & 0 & 0 & \dots & 0 & 0 \\ 0 & I & -Y_0^3(t_3) & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \dots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & I & -Y_0^{N_j}(t_{N_j}) \\ 0 & 0 & 0 & 0 & \dots & 0 & I \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_{N_j-1} \\ s_{N_j} \end{pmatrix} = \begin{pmatrix} v^2(t_2) \\ v^3(t_3) \\ \vdots \\ v^{N_j}(t_{N_j}) \\ y^2 \end{pmatrix} \tag{5.78}$$

After final time t^2 is reached, the linear system of Eq. (5.78) is assembled. As a result, vectors $s_j, j = 1, 2, \ldots, N_j$, are solved for and the solution $z(t)$ of the adjoint CPUODE is available on each sub-interval $\mathcal{I}^j = [t_j, t_{j+1}]$, $j = 1, 2, \ldots, N_j$,

$$z(t) = Y_0^j(t)s_j + v^j(t), t \in \mathcal{I}^j \tag{5.79}$$

It should be noted that although the matrix of the linear system of Eq. (5.78) can be very large, with dimension $N_{[t^1,t^2]} \times N_{[t^1,t^2]}$, where

$$N_{[t^1,t^2]} = 2d \cdot N_j \tag{5.80}$$

increases with the number N_j of fine-grid sub-intervals, it is an upper-triangular banded matrix in which the size of the upper-band; i.e., the largest non-zero component vector, on each column above the diagonal, does not exceed $4d$. Therefore, storage of the matrix requires only storage of the upper-band component vector of each column and the solution of Eq. (5.78) is obtained by backward substitution; i.e., the matrix of the linear system of Eq. (5.78) is already factored.

Since the piecewise solution defined by Eqs. (5.32) and (5.33) is $y^j(t) = \begin{pmatrix} \mu^v(t)^\top & \mu^{v\prime}(t)^\top \end{pmatrix}^\top$, for $t \in \mathcal{I}^j$, the independent adjoint variable $\mu^v(t)$ and its derivative $\mu^{v\prime}(t)$ at time $t \in \mathcal{I}^j$ are

$$\mu^v(t) = \begin{pmatrix} I_d & 0 \end{pmatrix} y^j(t) \tag{5.81}$$

$$\mu^{v\prime}(t) = \begin{pmatrix} 0 & I_d \end{pmatrix} y^j(t) \tag{5.82}$$

Substituting for the independent adjoint variable $\mu^v(t)$ of Eq. (5.81) into Eq. (5.25), the adjoint variable is evaluated as a function of $y^j(t)$,

$$\mu(t) = P_c X_0 \mu^v + P_c X_1 P_r r_a = P_c X_0 \begin{pmatrix} I_d & 0 \end{pmatrix} y^j(t) + P_c X_1 P_r r_a \tag{5.83}$$

where I_d is the $d \times d$ identity matrix. Substituting for the independent adjoint variable $\mu^v(t)$ of Eq. (5.81) and for its derivative of Eq. (5.82) into Eq. (5.27), the derivative of the adjoint variable $\mu(t)$ is evaluated as a function of $y^j(t)$,

$$\begin{aligned}
\mu'(t) &= P_c X_0 \mu^{v\prime} + P_c X_2 X_0 \mu^v + P_c (X_1 P_r r_a)' \\
&= \begin{pmatrix} P_c X_2 X_0 & P_c X_0 \end{pmatrix} \begin{pmatrix} \mu^v \\ \mu^{v\prime} \end{pmatrix} + P_c (X_1 P_r r_a)' \\
&= \begin{pmatrix} P_c X_2 X_0 & P_c X_0 \end{pmatrix} y^j(t) + P_c (X_1 P_r r_a)' \qquad (5.84)
\end{aligned}$$

Substituting for the piecewise solution $y^j(t)$ of the adjoint CPUODE at time $t \in \mathcal{I}^j$ its expression of Eq. (5.39) into Eqs. (5.83) and (5.84),

$$\mu(t) = P_c X_0 \begin{pmatrix} I_d & 0 \end{pmatrix} \left(Y_0^j(t) s_j + v^j(t) \right) + P_c X_1 P_r r_a \qquad (5.85)$$

$$\mu'(t) = \begin{pmatrix} P_c X_2 X_0 & P_c X_0 \end{pmatrix} \left(Y_0^j(t) s_j + v^j(t) \right) + P_c (X_1 P_r r_a)' \qquad (5.86)$$

the adjoint variable and its derivative are evaluated at time $t \in \mathcal{I}^j$ as linear functions of the fundamental matrix $Y_0^j(t)$, the particular solution $v^j(t)$, and constant vector s_j.

The adjoint acceleration constraint of Eq. (4.190),

$$\Phi_q \mu'' + 2 {\Phi_q}' \mu' + {\Phi_q}'' \mu = {r_a}'' \qquad (5.87)$$

together with Eq. (5.3), form the linear system

$$\begin{pmatrix} M & {\Phi_q}^\top \\ \Phi_q & 0 \end{pmatrix} \begin{pmatrix} \mu'' \\ \nu \end{pmatrix} = \begin{pmatrix} s_a \\ {r_a}'' \end{pmatrix} - \begin{pmatrix} D_2 & D_1 \\ {\Phi_q}'' & 2{\Phi_q}' \end{pmatrix} \begin{pmatrix} \mu \\ \mu' \end{pmatrix} \qquad (5.88)$$

with unknowns μ'' and ν. The matrix

$$M^S = \begin{pmatrix} M & {\Phi_q}^\top \\ \Phi_q & 0 \end{pmatrix} \tag{5.89}$$

is the Schur complement of the system and is non-singular [52]. Therefore, the vector $\nu(t)$ of adjoint Lagrange multipliers is obtained by solving the linear system of Eq. (5.88) and extracting the last m elements of the solution,

$$\nu(t) = \begin{pmatrix} 0 & I_m \end{pmatrix} {M^S}^{-1} \left(\begin{pmatrix} s_a \\ {r_a}'' \end{pmatrix} - H \begin{pmatrix} \mu \\ \mu' \end{pmatrix} \right) \tag{5.90}$$

where

$$H = \begin{pmatrix} D_2 & D_1 \\ {\Phi_q}'' & 2{\Phi_q}' \end{pmatrix} \tag{5.91}$$

Substituting for adjoint variable $\mu(t)$ and its derivative $\mu'(t)$ of Eqs. (5.85) and (5.86) into Eq. (5.90), $\nu(t)$ is re-written as

$$\nu(t) = \begin{pmatrix} 0 & I_m \end{pmatrix} {M^S}^{-1} \left(\begin{pmatrix} s_a \\ {r_a}'' \end{pmatrix} - H\Lambda \right) \tag{5.92}$$

where

$$\Lambda = \begin{pmatrix} P_0\big(Y_0^j(t)s_j + v^j(t)\big) + P_c X_1 P_r r_a \\ P_X\big(Y_0^j(t)s_j + v^j(t)\big) + P_c (X_1 P_r r_a)' \end{pmatrix} \tag{5.93}$$

and

$$\begin{aligned} P_0 &= P_c X_0 \begin{pmatrix} I_d & 0 \end{pmatrix} \\ P_X &= \begin{pmatrix} P_c X_2 X_0 & P_c X_0 \end{pmatrix} \end{aligned}$$

Therefore, the adjoint variable $\mu(t)$ and the vector of Lagrange multipliers $\nu(t)$ at time $t \in \mathcal{I}^j$ are functions of the following:

1. Permutation matrices P_c and P_r resulting from the foregoing partitioning,

$$P_r \Phi_q P_c = \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix}$$

on the fine-grid $\mathcal{I}^j$ and matrices X_0, X_1, D_1, D_2, Φ_q, $\Phi_q{}'$, $\Phi_q{}''$, and M that are computed at time t using multibody system state information $q(t)$, $q'(t)$, $q''(t)$, and $\lambda(t)$, obtained by interpolating for q, q', q'', and λ at the required time $t \in [t_j, t_{j+1}]$ using the previously stored $q(t_k)$, $q'(t_k)$, $q''(t_k)$, $t_k \in \Delta^j$.

2. The fundamental matrix $Y_0^j(t)$ that is obtained by applying the integrate-recover-assemble algorithm on interval $\mathcal{I}^j$ to the homogeneous adjoint DAE, with initial conditions of the form of Eqs. (5.68) and (5.69).

3. The particular solution $v^j(t)$ that is obtained as a result of applying the integrate-recover-assemble algorithm on interval $\mathcal{I}^j$ to the non-homogeneous adjoint DAE, with initial conditions of the form of Eqs. (5.70) and (5.71).

4. Vector s_j, which is available after t^2 is reached in solving the linear system of Eq. (5.78).

Separating data that is available at the end of the fine-grid $\mathcal{I}^j$ from data that is available after final time t^2 is reached, the adjoint variable $\mu(t)$ on the fine grid $\mathcal{I}^j$ is re-written as

$$\mu(t) = a_j^{\mu}(t) + B_j^{\mu}(t) s_j \tag{5.94}$$

where vector

$$a_j^{\mu}(t) = P_c X_0 \begin{pmatrix} I_d & 0 \end{pmatrix} v^j(t) + P_c X_1 P_r r_a \tag{5.95}$$

and matrix

$$B_j^{\mu}(t) = P_c X_0 \begin{pmatrix} I_d & 0 \end{pmatrix} Y_0^j(t) \tag{5.96}$$

are evaluated at each time t inside the fine-grid sub-interval $[t_j, t_{j+1}]$, while the evaluation of s_j is postponed until after t^2 is reached. Similarly, by separating data that is available at the end of the fine-grid $\mathcal{I}^j$ from data that is available after final time t^2 is reached, the vector of adjoint Lagrange multipliers $\nu(t)$ on the fine grid $\mathcal{I}^j$ is re-written as

$$\nu(t) = a_j^{\nu}(t) + B_j^{\nu}(t) s_j \tag{5.97}$$

where vector $a_j^{\nu}(t)$ is defined as

$$a_j^{\nu}(t) = \begin{pmatrix} 0 & I_m \end{pmatrix} M^{S^{-1}} \left(\begin{pmatrix} s_a \\ r_a'' \end{pmatrix} - H\Lambda_0 \right) \tag{5.98}$$

matrix $B_j^{\nu}(t)$ is defined as

$$B_j^{\nu}(t) = -\begin{pmatrix} 0 & I_m \end{pmatrix} M^{S^{-1}} H \begin{pmatrix} \Lambda_1 Y_0^j(t) \\ \Lambda_2 Y_0^j(t) \end{pmatrix} \tag{5.99}$$

and

$$\Lambda_0 = \begin{pmatrix} P_c X_1 P_r r_a + \Lambda_1 v^j(t) \\ P_c (X_1 P_r r_a)' + \Lambda_2 v^j(t) \end{pmatrix} \tag{5.100}$$

$$\Lambda_1 = P_c X_0 \begin{pmatrix} I_d & 0 \end{pmatrix} \tag{5.101}$$

$$\Lambda_2 = \begin{pmatrix} P_c X_2 X_0 & P_c X_0 \end{pmatrix} \tag{5.102}$$

Consequently, provided that the multibody system state vectors $q(t_k)$, $q'(t_k)$, $q''(t_k)$, and $\lambda(t_k)$ have been stored for each $t_k \in \Delta^j$, vectors $a_j^\mu(t)$ and $a_j^\nu(t)$ and matrices $B_j^\mu(t)$ and $B_j^\nu(t)$ can be evaluated, using Eqs. (5.95),(5.98),(5.96), and (5.99), at each time t inside the fine grid $[t_j, t_{j+1}]$, while evaluation of s_j is postponed until after t^2 is reached.

5.3 Evaluation of Gradients of Functionals Using Vectors a_j^μ and a_j^ν and Matrices B_j^μ and B_j^ν

The adjoint formulation for the gradient of the functional of Eq. (5.1) is [26]

$$\begin{aligned}\Psi_\beta &= l_\beta(t^2) + \mu^\top(t^1)K_1 - {\gamma^2}^\top\Phi_\beta(t^2) - {\eta^2}^\top(\Phi_q\hat{q}')_\beta(t^2) - \xi^2\Omega_\beta(t^2) \\ &- g_{q'}(t^1)q_\beta(t^1) - {\mu'}^\top(t^1)M(t^1)q_\beta(t^1) + \int_{t^1}^{t^2} g_\beta(t)dt \\ &- \int_{t^1}^{t^2}(\mu^\top(t)K_2(t) + \nu^\top(t)\Phi_\beta(t))dt\end{aligned} \tag{5.103}$$

where

$$K_1 = \big(Mq_\beta{}' - (S_{q'}^1 + M')q_\beta\big)(t^1) \tag{5.104}$$

and

$$K_2(t) = (M\hat{q}'')_\beta + ({\Phi_q}^\top\hat{\lambda})_\beta - S_\beta^1 \tag{5.105}$$

Vectors η^2 and γ^2 and scalar ξ^2 are evaluated after final time t^2 is reached, as defined in Ref. [26]. Assuming the coarse-grid interval $[t^1, t^2]$ is partitioned into fine-grid subintervals, $\mathcal{I}^j = [t_j, t_{j+1}]$, $j = 1, 2, \ldots, N_j$, where $t_1 \equiv t^1$ and $t_{N_j+1} \equiv t^2$, the gradient

Ψ_β of functional Ψ is a summation of the following terms:

1. One term that is evaluated at initial time t^1, $g_{q'}(t^1)q_\beta(t^1)$.

2. Terms that are evaluated after final time t^2 is reached; e.g., $l_\beta(t^2)$, ${\gamma^2}^\top\Phi_\beta(t^2)$, ${\eta^2}^\top(\Phi_q\hat{q}')_\beta(t^2)$, and $\xi^2\Omega_\beta(t^2)$.

3. One term that is incrementally evaluated on each fine-grid $[t_j, t_{j+1}]$, $\int_{t^1}^{t^2} g_\beta(t)dt$.

4. Partially postponed terms; i.e., terms of the form

$$x_j(t) = a_j(t) + B_j(t)s_j \tag{5.106}$$

where $a_j(t)$ and $B_j(t)$ can be evaluated at each time t inside the fine-grid $[t_j, t_{j+1}]$, and s_j can be evaluated only after t^2 is reached. Such terms are $\mu^\top(t_1)K_1$ and ${\mu'}^\top(t^1)M(t^1)q_\beta(t^1)$. Substituting for $t = t_1$ and $j = 1$ into Eq. (5.94),

$$\mu(t^1) = a_1^\mu(t^1) + B_1^\mu(t^1)s_1 \tag{5.107}$$

and substituting for for $t = t_1$ and $j = 1$ into Eq. (5.86),

$$\begin{aligned}
\mu'(t^1) &= \begin{pmatrix} P_c(X_2X_0)(t^1) & P_cX_0(t^1) \end{pmatrix} v^1(t^1) + P_c(X_1P_rr_a)'(t^1) \\
&+ \begin{pmatrix} P_c(X_2X_0)(t^1) & P_cX_0(t^1) \end{pmatrix} Y_0^1(t^1)s_1 \\
&= a_1^{\mu'}(t^1) + B_1^{\mu'}(t^1)s_1
\end{aligned} \tag{5.108}$$

where

$$a_j^{\mu'}(t) \equiv \begin{pmatrix} P_c(X_2X_0)(t) & P_cX_0(t) \end{pmatrix} v^j(t) + P_c(X_1P_rr_a)'(t) \tag{5.109}$$

$$B_j^{\mu'}(t) \equiv \begin{pmatrix} P_c(X_2X_0)(t) & P_cX_0(t) \end{pmatrix} Y_0^j(t) \tag{5.110}$$

$a_j^{\mu}(t)$ and $B_j^{\mu}(t)$ defined by Eqs. (5.95) and (5.96), are evaluated at initial time t^1; and vector s_1, which is a component of solution of the linear system of Eq. (5.78), is evaluated only after final time t^2 is reached.

5. Integrals of partially postponed terms; i.e., terms of the form

$$\int_{t^1}^{t^2} {x_j}^{\top}(t)X(t)dt = \sum_{j=1}^{N_j} \int_{t_j}^{t_{j+1}} x^{\top}(t)X(t)dt \tag{5.111}$$

where $x_j(t)$ is a partially postponed vector of the form of Eq. (5.106) and $X(t)$ is a matrix that can be evaluated at each time t inside the fine-grid $[t_j, t_{j+1}]$. Therefore,

$$\begin{aligned}
\int_{t^1}^{t^2} x^{\top}(t)X(t)dt &= \sum_{j=1}^{N_j} \int_{t_j}^{t_{j+1}} \left({a_j}^{\top}(t) + {s_j}^{\top}{B_j}^{\top}(t)\right)X(t)dt \\
&= \sum_{j=1}^{N_j} \left(\int_{t_j}^{t_{j+1}} {a_j}^{\top}(t)X(t)dt + {s_j}^{\top} \int_{t_j}^{t_{j+1}} {B_j}^{\top}(t)X(t)dt \right) \\
&= \sum_{j=1}^{N_j} \int_{t_j}^{t_{j+1}} {a_j}^{\top}(t)X(t)dt \\
&+ \sum_{j=1}^{N_j} {s_j}^{\top} \int_{t_j}^{t_{j+1}} {B_j}^{\top}(t)X(t)dt = \sum_{j=1}^{N_j} \tau_j + \sum_{j=1}^{N_j} {s_j}^{\top} T_j
\end{aligned} \tag{5.112}$$

where vector

$$\tau_j = \int_{t_j}^{t_{j+1}} {a_j}^{\top}(t)X(t)dt \tag{5.113}$$

and matrix

$$T_j = \int_{t_j}^{t_{j+1}} {B_j}^{\top}(t)X(t)dt \tag{5.114}$$

are evaluated at the end of each fine-grid $[t_j, t_{j+1}]$ and stored until after final time t^2 is reached, when vectors s_j, $j = 1, 2, \ldots, N_j$, are computed by solving the linear system

of Eq. (5.78). After final time t^2 is reached, the integral of Eq. (5.111) is assembled according to Eq. (5.112). It should be noted that, while matrices T_j need to be stored at the end of each sub-interval $[t_j, t_{j+1}]$, $j = 1, 2, \ldots, N_j$, only the sum $\tau = \sum_{j=1}^{N_j} \tau_j$ of vectors τ_j needs to be stored for assembling the integral of Eq. (5.111).

5.3.1 Evaluation of the Initial Time Term

Evaluation of the term $g_{q'}(t^1)q_\beta(t^1)$ requires initial conditions $q(t^1)$, $v(t^1)$, $a(t^1)$, and $\lambda(t^1)$ of Eqs. (4.15) and (4.16) and the initial sensitivity matrix $q_\beta(t^1)$. The initial sensitivity

$$q_\beta(t^1) = \begin{pmatrix} q_{\beta_1}(t^1) & q_{\beta_2}(t^1) & \ldots & q_{\beta_{n_\beta}}(t^1) \end{pmatrix}$$

is obtained by solving for the initial conditions of the direct differentiation DAE, as shown in Section 4.3 of Chapter 4.

5.3.2 Evaluation of Final Time Terms

After final time t^2 is reached; i.e., when $\Omega(t, q, q', \beta) = 0$, terms $l_\beta(t^2)$, ${\gamma^2}^\top \Phi_\beta(t^2)$, ${\eta^2}^\top (\Phi_q \hat{q}')_\beta(t^2)$, and $\xi^2 \Omega_\beta(t^2)$ are computed as follows:

1. Term $l_\beta(t^2)$ is evaluated directly, since $l(t^2, q^2, q'^2, \beta)$ is given.

2. Terms depending on γ^2, η^2, and ξ^2 are evaluated by solving first for the initial conditions of the adjoint DAE, as shown in Section 4.4 of Chapter 4. After vectors γ^2 and η^2 and scalar ξ^2 are computed, $\Phi_\beta(t^2)$ and $(\Phi_q \hat{q}')_\beta(t^2)$ are computed, using the procedure presented in Section 4.3.2 of Chapter 4.

3. Term $\Omega_\beta(t^2)$ is evaluated directly, since $\Omega(t, q, q', \beta)$ is given.

5.3.3 Incremental Evaluation of Integral $\int_{t^1}^{t^2} g_\beta(t)dt$

As a result of integration of the equations of the motion on fine-grid subinterval $\mathcal{I}^j$, the generalized coordinate vector $q(t)$, its derivatives $q'(t)$ and $q''(t)$, and the Lagrange multiplier $\lambda(t)$ are computed and stored on the fine-grid mesh Δ^j. Function $g(q, q', \lambda, \beta, t)$ is given and vectors $q(t), q'(t)$, and $\lambda(t)$ are interpolated at each point inside the fine-grid $[t_j, t_{j+1}]$, using previously stored $q(t_k)$, $q'(t_k)$, $q''(t_k)$, and $\lambda(t_k)$, for each $t_k \in \Delta^j$. As a result, the integral

$$I^g_{j,0} = \int_{t_j}^{t_{j+1}} g_\beta(t)dt \tag{5.115}$$

is computed at the end of the fine-grid, using a quadrature numerical formula; e.g., a Newton-Cotes or Gauss [8] integration formula. The result is then added to the sum of integrals of $g_\beta(t)$ on previous intervals, using the recurrence formula

$$I^g_{j+1} = I^g_j + I^g_{j,0},\ j = 1, 2, \ldots, N_j - 1 \tag{5.116}$$

where $I^g_1 = I^g_{1,0} = \int_{t_1}^{t_2} g_\beta(t)dt$ is the integral of $g_\beta(t)$ on the first fine-grid $[t_j, t_{j+1}]$ with $j = 1$. After final time t^2 is reached, the recurrence relation of Eq. (5.116) yields

$$I^g_{N_j+1} = \sum_{j=1}^{N_j} \int_{t_j}^{t_{j+1}} g_\beta(t)dt = \int_{t^1}^{t^2} g_\beta(t)dt \tag{5.117}$$

5.3.4 Evaluation of Partially Postponed Terms

After final time t^2 is reached, the linear system of Eq. (5.78) is solved and vectors s_j, $j = 1, 2, \ldots, N_j$, are available. As a result, the adjoint variable $\mu(t^1)$ and

its derivative $\mu'(t^1)$ are evaluated by substituting for vector s_1 into Eqs. (5.107) and (5.108), in which fundamental matrix $Y_0^1(t^1)$ and particular solution $v^1(t^1)$ have been previously stored at the end of the first fine-grid $[t_1, t_2]$. Matrix K_1 defined by Eq. (5.104) requires initial sensitivities $q_\beta(t^1)$ and $q'_\beta(t^1)$, which are computed at initial time t^1, as shown in Section 5.3.1; matrices $M(t^1)$ and $S^1_{q'}(t^1)$, which are computed at initial time t^1, as shown in Sections 4.2.4 and 4.2.5 of Chapter 4; and the derivative of the mass matrix $M'(t^1)$, as shown in Section 4.4.2 of Chapter 4. As a result, $\mu^\top(t^1)K_1$ and $\mu'^\top(t^1)M(t^1)q_\beta(t^1)$ are assembled using the adjoint variable μ and its derivative μ' at initial time t^1, mass matrix $M(t^1)$, matrix K_1, and initial sensitivities $q_\beta(t^1)$.

5.3.5 Evaluation of Integrals of Partially Postponed Terms

Evaluation of the integral

$$I^\mu_{j,0} = \int_{t_j}^{t_{j+1}} \mu^\top(t) K_2(t) dt \tag{5.118}$$

is decomposed as follows:

$$\begin{aligned} I^\mu_{j,0} &= \int_{t_j}^{t_{j+1}} \left(a_j^{\mu\top}(t) + s_j{}^\top B_j^{\mu\top}(t)\right) K_2(t) dt \\ &= \int_{t_j}^{t_{j+1}} a_j^{\mu\top}(t) K_2(t) dt + s_j{}^\top \int_{t_j}^{t_{j+1}} B_j^{\mu\top}(t) K_2(t) dt \\ &= \tau^{\mu\top}_{j,0} + s_j{}^\top T_j^\mu \end{aligned} \tag{5.119}$$

where

$$\tau_{j,0}^{\mu\,\top} = \int_{t_j}^{t_{j+1}} a_j^{\mu\top}(t) K_2(t) dt \tag{5.120}$$

$$T_j^{\mu} = \int_{t_j}^{t_{j+1}} B_j^{\mu\top}(t) K_2(t) dt \tag{5.121}$$

and $a_j^{\mu\top}(t)$ and $B_j^{\mu\top}(t)$ are defined by Eqs. (5.95) and (5.96).

As a result,

$$\begin{aligned} I^{\mu} &= \int_{t^1}^{t^2} \mu^{\top}(t) K_2(t) dt = \sum_{j=1}^{N_j} \int_{t_j}^{t_{j+1}} \mu^{\top}(t) K_2(t) dt \\ &= \sum_{j=1}^{N_j} I_{j,0}^{\mu} = \sum_{j=1}^{N_j} (\tau_{j,0}^{\mu\,\top} + s_j{}^{\top} T_j^{\mu}) = \sum_{j=1}^{N_j} \tau_{j,0}^{\mu\,\top} + \sum_{j=1}^{N_j} s_j{}^{\top} T_j^{\mu} \end{aligned} \tag{5.122}$$

where $\sum_{j=1}^{N_j} \tau_{j,0}^{\mu\,\top}$ is incrementally computed using the recurrence formula

$$\tau_{j+1}^{\mu} = \tau_j^{\mu} + \tau_{j,0}^{\mu} \tag{5.123}$$

where $\tau_1^{\mu} = \tau_{1,0}^{\mu}$ and $\tau_{j,0}^{\mu}$ are computed using Eq. (5.120) on fine-grid $[t_j, t_{j+1}]$. Therefore, at final time $t^2 \equiv t_{N_j+1}$,

$$\tau_{N_j+1}^{\mu\,\top} = \sum_{j=1}^{N_j} \tau_{j,0}^{\mu\,\top} \tag{5.124}$$

Terms T_j^{μ}, $j = 1, 2, \ldots, N_j$, are computed using Eq. (5.121) on fine-grid $[t_j, t_{j+1}]$ and stored. After final time $t^2 \equiv t_{N_j+1}$ is reached and s_j, $j = 1, 2, \ldots, N_j$, are obtained, as a result of solving Eq. (5.78), the integral

$$I^{\mu} = \sum_{j=1}^{N_j} \tau_{j,0}^{\mu\,\top} + \sum_{j=1}^{N_j} s_j{}^{\top} T_j^{\mu} = \tau_{N_j+1}^{\mu\,\top} + \sum_{j=1}^{N_j} s_j{}^{\top} T_j^{\mu} \tag{5.125}$$

is assembled using incrementally computed term $\tau_{N_j+1}^{\mu}$, stored terms T_j^{μ}, $j = 1, 2, \ldots, N_j$, and vectors s_j. Both $\tau_{j,0}^{\mu}$ and T_j^{μ} require evaluation of matrix

$$K_2(t) = (M\hat{q}'')_{\beta} + (\Phi_q{}^{\top}\hat{\lambda})_{\beta} - S_{\beta}^1 \tag{5.126}$$

in which $(M\hat{q}'')_\beta$, $({\Phi_q}^\top\hat{\lambda})_\beta$, and S^1_β are evaluated at any point t inside the fine-grid $[t_j, t_{j+1}]$, using formulas developed in Section 4.3 of Chapter 4 and previously stored $q(t_k)$, $q'(t_k)$, $q''(t_k)$, and $\lambda(t_k)$, for each $t_k \in \Delta^j$. The numerical integration required by terms $\tau^\mu_{j,0}$ and T^μ_j is performed using a quadrature formula; e.g., Newton-Cotes or Gauss [8]. Similarly, evaluation of the integral

$$I^\nu_{j,0} = \int_{t_j}^{t_{j+1}} \nu^\top(t)\Phi_\beta(t)dt \tag{5.127}$$

is decomposed as follows:

$$I^\nu_{j,0} = \int_{t_j}^{t_{j+1}} {a^\nu_j}^\top(t)\Phi_\beta(t)dt + {s_j}^\top \int_{t_j}^{t_{j+1}} {B^\nu_j}^\top(t)\Phi_\beta(t)dt = {\tau^\nu_{j,0}}^\top + {s_j}^\top T^\nu_j \tag{5.128}$$

where ${a^\nu_j}^\top(t)$ and ${B^\nu_j}^\top(t)$ are defined by Eqs. (5.98) and (5.99), respectively, and

$$\begin{aligned} {\tau^\nu_{j,0}}^\top &= \int_{t_j}^{t_{j+1}} {a^\nu_j}^\top(t)\Phi_\beta(t)dt & (5.129)\\ T^\nu_j &= \int_{t_j}^{t_{j+1}} {B^\nu_j}^\top(t)\Phi_\beta(t)dt & (5.130) \end{aligned}$$

As a result,

$$\begin{aligned} I^\nu &= \int_{t^1}^{t^2} \nu^\top(t)\Phi_\beta(t)dt = \sum_{j=1}^{N_j} \int_{t_j}^{t_{j+1}} \nu^\top(t)\Phi_\beta(t)dt \\ &= \sum_{j=1}^{N_j} I^\nu_{j,0} = \sum_{j=1}^{N_j} {\tau^\nu_{j,0}}^\top + \sum_{j=1}^{N_j} {s_j}^\top T^\nu_j \end{aligned} \tag{5.131}$$

where $\sum_{j=1}^{N_j} {\tau^\nu_{j,0}}^\top$ is incrementally computed using the recurrence formula

$$\tau^\nu_{j+1} = \tau^\nu_j + \tau^\nu_{j,0} \tag{5.132}$$

where $\tau^\nu_1 = \tau^\nu_{1,0}$ and $\tau^\nu_{j,0}$ are computed using Eq. (5.129) on fine-grid $[t_j, t_{j+1}]$. Therefore, at final time $t^2 \equiv t_{N_j+1}$,

$${\tau^\nu_{N_j+1}}^\top = \sum_{j=1}^{N_j} {\tau^\nu_{j,0}}^\top \tag{5.133}$$

Terms T_j^ν, $j = 1, 2, \ldots, N_j$, are computed using Eq. (5.130) on fine-grid $[t_j, t_{j+1}]$ and then stored. After the final time is reached, the integral

$$I^\nu = \sum_{j=1}^{N_j} {\tau_{j,0}^\nu}^\top + \sum_{j=1}^{N_j} {s_j}^\top T_j^\nu = {\tau_{N_j+1}^\nu}^\top + \sum_{j=1}^{N_j} s_j T_j^\nu \tag{5.134}$$

is assembled using the incrementally computed term ${\tau_{N_j+1}^\nu}^\top$, stored terms T_j^ν, $j = 1, 2, \ldots, N_j$, and vectors s_j obtained by solving the linear system of Eq. (5.78). Both $\tau_{j,0}^\nu$ and T_j^ν require the evaluation of matrix Φ_β, which is computed at any time t inside the fine-grid $[t_j, t_{j+1}]$, using formulas developed in Section 4.3 of Chapter 4 and previously stored $q(t_k)$, $q'(t_k)$, $q''(t_k)$, and $\lambda(t_k)$, for each $t_k \in \Delta^j$.

As a result, the integral of partially postponed terms, on the coarse-grid interval $[t^1, t^2]$,

$$\begin{aligned} I^{\mu,\nu} &\equiv I^\mu + I^\nu = \int_{t^1}^{t^2} \mu^\top(t) K_2(t) dt + \int_{t^1}^{t^2} \nu^\top(t) \Phi_\beta(t) dt \\ &= \sum_{j=1}^{N_j} {\tau_{j,0}^\mu}^\top + \sum_{j=1}^{N_j} {s_j}^\top T_j^\mu \\ &+ \sum_{j=1}^{N_j} {\tau_{j,0}^\nu}^\top + \sum_{j=1}^{N_j} {s_j}^\top T_j^\nu \end{aligned} \tag{5.135}$$

which, by ordering terms and defining

$$\tau = \sum_{j=1}^{N_j} (\tau_{j,0}^\mu + \tau_{j,0}^\nu) \tag{5.136}$$

$$T_j = T_j^\mu + T_j^\nu \tag{5.137}$$

is re-written as

$$I^{\mu,\nu} = \tau^\top + \sum_{j=1}^{N_j} ({s_j}^\top T_j) \tag{5.138}$$

where τ is updated at the end of each fine-grid sub-interval $[t_j, t_{j+1}]$; T_j are computed using a quadrature formula on the fine-grid $[t_j, t_{j+1}]$, $j = 1, 2, \ldots, N_j$; and vectors s_j

are obtained by solving the linear system of Eq. (5.78), after final time t^2 is reached. Consequently, the *integral of partially postponed terms* algorithm, for computing the integral

$$I^{\mu,\nu} = \int_{t^1}^{t^2} (\mu^\top(t) K_2(t) + \nu^\top(t) \Phi_\beta(t)) dt \tag{5.139}$$

is defined as follows:

1. Initialization. Let $t = t^1$, $j = 1$, and $\tau = 0$. Compute initial conditions of Eqs. (4.15) and (4.16), $q_1 = q(t, \beta)$, $v_1 = q'(t, \beta)$, $a_1 = (''t, \beta)$ and $\lambda_1 = \lambda(t, \beta)$, as shown in Section 4.2 of Chapter 4. Let $q_{test} = q_1$.

2. Termination test. If $\|\Omega(t, q_{test}, \beta)\| < \epsilon$, let $t^2 = t$ and go to step 7. Else continue.

3. Integration of equations of motion . Define fine-grid sub-interval $\mathcal{I}^j = [t_j, t_{j+1}]$, where $t_j = t$ and $t_{j+1} = t + h_j$. Integrate Eqs. (4.15) and (4.16) and store $q(t_k)$, $q'(t_k)$, $q''(t_k)$, and $\lambda(t_k)$ solved for on the mesh $\Delta^j = \{t_{k_1} = t_j, \ldots, t_{k_2} = t_{j+1}\}$. Define $q_{test} = q(t_{j+1}, \beta)$.

4. Application of integrate-recover-assemble algorithm. The linearly independent columns $y_i^j(t)$, $i = 1, 2, \ldots, 2d$, of the fundamental matrix $Y_0^j(t)$ are obtained, at each $t_k \in \Delta^j$, by independently applying the integrate-recover-assemble algorithm presented in Section 5.2 to the homogeneous adjoint DAE of Eqs. (5.3) and (5.4), with initial conditions of the form of Eqs. (5.68) and (5.69). The particular solution $v^j(t)$ is independently obtained by applying the integrate-recover-assemble algorithm to the non-homogeneous adjoint DAE of Eqs. (5.3)

and (5.4), with initial conditions of the form of Eqs. (5.70) and (5.71).

5. Evaluation of τ and T_j. Applying quadrature formulas to ${a_j^{\mu}}^{\top}(t)K_2(t)$, ${a_j^{\nu}}^{\top}(t)\Phi_\beta(t)$, ${B_j^{\mu}}^{\top}(t)K_2(t)$, and ${B_j^{\nu}}^{\top}(t)\Phi_\beta(t)$, vectors $\tau_{j,0}^{\mu}$ and $\tau_{j,0}^{\nu}$, and matrices T_j^{μ} and T_j^{ν} of Eqs. (5.120), (5.129), (5.121), and (5.130) are obtained; τ of Eq. (5.136) is incrementally updated; and T_j of Eq. (5.137) is computed on the fine-grid $\mathcal{I}^j$ and stored.

6. Step advance. Let $j = j + 1$ and $t = t_{j+1}$. Go to step 2.

7. Finalization. Solve the linear system of Eq. (5.78) for vectors s_j, $j = 1, 2, \ldots, N_j$. Assemble the integral $I^{\mu,\nu}$ of partially postponed terms, from vectors s_j, previously stored T_j, and τ.

By comparison with evaluation of the gradient Ψ_β using the adjoint formulation [26] presented in Section 4.4 of Chapter 4, or the direct differentiation formulation [26] presented in Section 4.3 of Chapter 4, the advantages of computing gradients of functionals through the piecewise solution of the adjoint DAE; i.e., assembling the gradient Ψ_β by evaluating the initial time term, as shown in Section 5.3.1; incrementally evaluating the $\int_{t^1}^{t^2} g_\beta(t)dt$, as shown in Section 5.3.3; evaluating the partially postponed terms, as shown in Section 5.3.4; and evaluating the integrals of partially postponed terms, as shown above, follows mainly from the structure of the integral of partially postponed terms algorithm. They are as follows:

1. The fundamental matrix $Y_0^j(t)$ and particular solution $v^j(t)$ are independently evaluated using $2d + 1$ threads of computation.

2. Steps 3, 4, and 5 in the integral of partially postponed terms algorithm can be simultaneously performed in a three-stage pipeline; i.e., while integration of the equations of motion is performed on fine-grid sub-interval $\mathcal{I}^{j+1}$, application of the integrate-recover-assemble algorithm is performed on fine-grid $\mathcal{I}^{j}$, and evaluation of terms τ and T_j is performed on fine-grid $\mathcal{I}^{j-1}$.

3. In order to compute gradients of functionals through the direct differentiation formulation [26], independent integration of n_β index-3 DAE is required, while computing gradients of functionals through the piecewise solution of the adjoint DAE requires independent integration of only $2d+2$ DAE; one DAE integration for the equations of motion, $2d$ integrations for the $2d$ columns of the fundamental matrix $Y_0^j(t)$, and one integration for the particular solution $v^j(t)$.

4. The adjoint formulation [26] requires integration of only two DAE, the equations of motion and the adjoint DAE, but their integration must be performed sequentially; i.e., integration of the adjoint DAE can start only after integration of equations of motion has reached final time t^2. By contrast, computation of gradients of functionals through piecewise solution of the adjoint DAE progresses forward in time, along with integration of the equations of motion, requiring the additional evaluation of partially postponed terms and assembly of integrals of partially postponed terms by solving the linear system of Eq. (5.78), after final time t^2 is reached.

5. The fundamental matrix $Y_0^j(t)$ can be used for computation of gradients of many

functionals, since it is obtained by solving the homogeneous adjoint DAE, as shown by the integrate-recover-assemble algorithm. Therefore, it is not affected by the right-side of the adjoint DAE, which depends [26] on the structure of the functionals.

The disadvantage of computing gradients of functionals through piecewise solution of the adjoint DAE is that, in addition to the adjoint DAE integrations resulting from the integrate-recover-assemble algorithm, the constraint Jacobian must be factored. Its factorization is required for the following:

1. evaluation of vectors $a_j^\mu(t)$ and $a_j^\nu(t)$ and matrices $B_j^\mu(t)$ and $B_j^\nu(t)$, defined by Eqs. (5.95),(5.98),(5.96), and (5.99).

2. recovery of the independent adjoint variable and its derivative, defined by Eqs. (5.15) and (5.16).

5.4 Algorithms for Evaluating Gradients of Functionals Through The Direct Differentiation, Adjoint, and Piecewise Adjoint Methods

In this Section, outlines of the algorithms for the Direct Differentiation, Adjoint, and Piecewise Adjoint methods are presented. Interpolation methods for interpolating the equation of motion variables q, $v \equiv q'$, $a \equiv q''$, and λ, which are used

for constructing and solving the index-1 Direct Differentiation DAE of (4.113), the index-1 Adjoint DAE of (4.194), and the Adjoint CPUODE of Eq. (5.31), are chosen depending on the number of interpolating nodes and the availability of derivatives, as follows [8]:

1. Functions for which derivatives are not available; e.g., $a(t)$ and $\lambda(t)$, are interpolated using Lagrange interpolation with divided differences if the number of interpolating nodes is larger than ten, or Natural Splines if the number of interpolating nodes is between four and ten.

2. Functions for which first derivatives are available; e.g., $q(t)$ and $v(t)$ with derivatives $v(t)$ and $a(t)$, respectively, are interpolated using Complete Splines or Cubic Hermite, depending on the number of interpolating nodes. A Complete Splines interpolant is used when the number of interpolating nodes is larger than three. A Cubic Hermite interpolant is used when the number of interpolating nodes is two. A pair of Cubic Hermite interpolants is used when the number of interpolating nodes is three.

Figures 5.1 and 5.2 present the algorithm for evaluating gradients of functionals through the Direct Differentiation method. The gradient of functional Ψ of Eq. (4.181), in the Direct Differentiation formulation, is [26]

$$\begin{aligned}
\Psi_\beta &= (l_q q_\beta + l_{q'} q'_\beta)(t^2) + l_\beta(t^2) \\
&- (l_t + g + l_q q' + l_{q'} q'') \frac{(\Omega_q q_\beta + \Omega_\beta)}{\Omega'}(t^2) \\
&+ \int_{t^1}^{t^2} (g_q q_\beta + g_{q'} q'_\beta + g_\lambda \lambda_\beta + g_\beta)(t) dt
\end{aligned} \tag{5.140}$$

The term

$$\begin{aligned} g^2 &= (l_q q_\beta + l_{q'} q'_\beta)(t^2) + l_\beta(t^2) \\ &- (l_t + g + l_q q' + l_{q'} q'') \frac{(\Omega_q q_\beta + \Omega_\beta)}{\Omega'}(t^2) \end{aligned} \tag{5.141}$$

is evaluated after final time t^2 is reached. Figure 5.3 presents the algorithms for the Adjoint method. Figures 5.4 through 5.6 present the implemented algorithm for the Piecewise Adjoint method.

Both the Direct Differentiation and the Piecewise Adjoint method can benefit from a pipelined structure of the algorithm. In the Direct Differentiation method the following pipeline stages can be independently performed:

1. Pipeline stage 1. The integration of the DAE of motion and temporary storage of the equation of motion states on fine-grid $\mathcal{I}^{j+1}$.

2. Pipeline stage 2. The integration of the Direct Differentiation DAE, for each parameter $\beta_l, l = 1, 2, \ldots, n_\beta$ on fine-grid $\mathcal{I}^j$, using the equation of motion states previously stored by pipeline stage 1.

3. Pipeline stage 3. The integration of the gradient of functional on fine-grid $\mathcal{I}^{j-1}$ using the information previously evaluated and stored by pipeline stages 1 and 2.

In the Piecewise Adjoint method the following pipeline stages can be independently performed:

1. Pipeline stage 1. The integration of the DAE of motion and temporary storage of the equation of motion states on fine-grid $\mathcal{I}^{j+1}$.

2. Pipeline stage 2. The integration of the particular solution and columns of the fundamental matrix of the Adjoint CPUODE on fine-grid $\mathcal{I}^j$, using the equation of motion states previously stored by pipeline stage 1.

3. Pipeline stage 3. The integration of the gradient of functional on fine-grid $\mathcal{I}^{j-1}$ using the information previously evaluated and stored by pipeline stages 1 and 2.

Figures 5.7 and 5.8 present the pipelined version of the Piecewise Adjoint method.

5.5 Efficiency Analysis

This Section presents an estimation of the number of floating-point arithmetic operations (flops) for the Piecewise Adjoint method. Since floating-point additions are less expensive than floating-point multiplications or divisions [8], only multiplications and divisions are counted. The implemented parallel version of Figs. 5.4 through 5.6, the pipelined parallel version of Figs. 5.7 and 5.8, and the implemented sequential version are analyzed. The algorithm of the implemented sequential Piecewise Adjoint method is obtained from the algorithm of pipelined parallel version of Figs. 5.7 and 5.8 by running the pipeline stages sequentially.

5.5.1 Floating Point Operation Estimates

The most expensive computing effort of the Piecewise Adjoint method is spent inside the main loop; i.e., the *Do while* loop in Figs. 5.7 and 5.8 that advances the integration of the gradient of functional Ψ_β from the initial time t^1 to the final time

Let $t_1 = t^1$; $j = 1$;

For $l = 1, 2, \ldots, n_\beta$

$I_{q_l} = I_{q'_l} = I_{\lambda_l} = I_{\beta_l} = 0$;

End For

Evaluate IC

$q(\beta, t_1), q'(\beta, t_1), q''(\beta, t_1), \lambda(\beta, t_1)$;

$q_\beta(\beta, t_1), q'_\beta(\beta, t_1), q''_\beta(\beta, t_1), \lambda_\beta(\beta, t_1)$;

Do while $|\Omega(t_j, q(\beta, t_j), \beta)| > \epsilon$

pipeline stage 1

define *subinterval* $\mathcal{I}^j = [t_j, t_{j+1}]$ *and* **fine-grid**

$\Delta^j = \{t_{k_1} = t_j, \ldots, t_{k_2} = t_{j+1}\}$;

solve *the DAE of motion of Eq.* (4.20) *on* Δ^j;

store $\{q(\beta, t_k), q'(\beta, t_k), q''(\beta, t_k), \lambda(\beta, t_k), t_k \in \Delta^j\}$;

End pipeline stage 1

pipeline stage 2

For $l = 1, 2, \ldots, n_\beta$

independently *solve the DAE of Eq.* (4.113) *for parameter* β_l;

store $\{q_{\beta_l}(t_k), q'_{\beta_l}(t_k), \lambda_{\beta_l}(t_k), t_k \in \Delta^j\}$;

End For

End pipeline stage 2

Figure 5.1: The algorithm for the evaluation of gradients of functionals through the Direct Differentiation method

pipeline stage 3

update $I_{q_l} = I_{q_l} + \int_{t_j}^{t_{j+1}} g_q q_{\beta_l} dt$;

update $I_{q'_l} = I_{q'_l} + \int_{t_j}^{t_{j+1}} g_{q'} q'_{\beta_l} dt$;

update $I_{\lambda_l} = I_{\lambda_l} + \int_{t_j}^{t_{j+1}} g_\lambda \lambda_{\beta_l} dt$;

update $I_{\beta_l} = I_{\beta_l} + \int_{t_j}^{t_{j+1}} g_\beta(t) dt$;

End pipeline stage 3

Let $j = j + 1$;

End Do

For $l = 1, 2, \ldots, n_\beta$

evaluate final term g_l;

assemble $\Psi_{\beta_l} = g_l^2 + I_{q_l} + I_{q'_l} + I_{\lambda_l} + I_{\beta_l}$;

End For

Figure 5.2: The algorithm for the evaluation of gradients of functionals through the Direct Differentiation method-continued

Let $t_1 = t^1$; $j = 1$;

Evaluate IC

$q(\beta, t_1), q'(\beta, t_1), q''(\beta, t_1), \lambda(\beta, t_1)$;

$q_\beta(\beta, t_1), q'_\beta(\beta, t_1), q''_\beta(\beta, t_1), \lambda_\beta(\beta, t_1)$;

Evaluate $g_1 = g_{q'}(t^1) q_\beta(t^1)$;

Do while $|\Omega(t_j, q(\beta, t_j), \beta)| > \epsilon$

$t_k = t_j + h$;

solve *the DAE of motion of Eq.* (4.20)

store $\{q(\beta, t_k), q'(\beta, t_k), q''(\beta, t_k), \lambda(\beta, t_k)\}$;

$t_j = t_k$;

End Do

Evaluate *Adjoint IC of Eqs.* (4.197) *through* (4.198);

Evaluate *the* **final terms** $l_\beta(t^2), {\gamma^2}^\top \Phi_\beta(t^2), {\eta^2}^\top (\Phi_q \hat{q}')_\beta(t^2)$, *and* $\xi^2 \Omega_\beta(t^2)$;

Do while $t_j > t^1$

solve *the Adjoint DAE of Eq.* (4.194),

by interpolating *for* $\{q(\beta, t_j), q'(\beta, t_j), q''(\beta, t_j), \lambda(\beta, t_j)\}$ *using the stored set*

$\{q(\beta, t_k), q'(\beta, t_k), q''(\beta, t_k), \lambda(\beta, t_k), t_k = t^1, \ldots, t^2\}$;

update $I_g = I_g + \int_{t_j - h}^{t_j} g_\beta(t) dt$;

update $I^{\mu,\nu}$ *of Eq.* (5.139);

$t_j = t_j - h$;

End Do

Evaluate $\mu(t^1), \mu'(t^1)$ *of Eqs.* (5.107) *and* (5.108);

Evaluate *the* **initial terms** ${\mu}^\top(t^1) k_1$ *and* ${\mu'}^\top(t^1) M(t^1) q_\beta(t^1)$;

Evaluate *the gradient of the functional of Eq.* (5.103), *by assembling* $I_g, I^{\mu,\nu}$,

the **initial terms**,*the* **final terms**, and g_1;

Figure 5.3: The algorithm for the evaluation of gradients of functionals through the Adjoint method

Let $t_1 = t^1$; $j = 1$;

Evaluate *IC*

$q(\beta, t_1), q'(\beta, t_1), q''(\beta, t_1), \lambda(\beta, t_1)$;

$q_\beta(\beta, t_1), q'_\beta(\beta, t_1), q''_\beta(\beta, t_1), \lambda_\beta(\beta, t_1)$;

Evaluate $g_1 = g_{q'}(t^1)q_\beta(t^1)$; $\tau = 0$; $I_g = 0$;

Create $2d + 1$ **ChildrenThreads** *in addition to the* **MainThread**;

Do while $|\Omega(t_j, q(\beta, t_j), \beta)| > \epsilon$

Set the $2d + 1$ **ChildrenThreads** on **WAIT** state;

MainThread Executes:

define *subinterval* $\mathcal{I}^j = [t_j, t_{j+1}]$ *and* **fine-grid**

$\Delta^j = \{t_{k_1} = t_j, \ldots, t_{k_2} = t_{j+1}\}$;

solve *the DAE of motion of Eq.* (4.20) *on* Δ^j;

store $\{q(\beta, t_k), q'(\beta, t_k), q''(\beta, t_k), \lambda(\beta, t_k), t_k \in \Delta^j\}$;

Set *the* **ChildrenThreads** *on* **READY** *state*;

Set *the* **MainThread** *on* **WAIT** *state*;

Each **ChildThread Executes**:

independently *compute the LU factorization of* Φ_q *at each mode* $t_k \in \Delta^j$;

Notify MainThread *that* **ChildThread** *is* **DONE**;

Set ChildThread *on* **WAIT** state;

After all **ChildrenThreads Notify** *the* **MainThread**

Set *the* **MainThread** *on* **READY** state;

MainThread Executes:

store *the factored constraint Jacobians* $\{\Phi_q^{LU}(t_k), t_k \in \Delta^j\}$;

Set *the* **ChildrenThreads** *on* **READY** state;

Set *the* **MainThread** *on* **WAIT** state;

Figure 5.4: The implemented parallel algorithm of the Piecewise Adjoint method

Each **ChildThread Executes**:

by **interpolating** *the set*

$\{q(\beta,t_k), q'(\beta,t_k), q''(\beta,t_k), \lambda(\beta,t_k), t_k \in \Delta^j\}$ *and using the set*

$\{\Phi_q^{LU}, t_k \in \Delta^j\}$,

independently *evaluate a column of the fundamental matrix*

$\{Y_0{}^j(t_k), t_k \in \Delta^j\}$ *or the particular solution* $\{v^j(t_k), t_k \in \Delta^j\}$,

through the **integrate-recover-assemble** *algorithm, defined*

in Section 5.2;

store $Y_0{}^j(t_j), v^j(t_j)$;

Notify MainThread *that* **ChildThread** *is* **DONE**;

Set ChildThread on **WAIT** state;

After all **ChildrenThreads Notify** *the* **MainThread**

Set *the* **MainThread** *on* **READY** state;

MainThread Executes:

using $\{Y_0^j(t_k), v^j(t_k), t_k \in \Delta^j\}$ **update** τ *of Eq.* (5.136), T_j *of Eq.* (5.137);

update $I_g = I_g + \int_{t_j}^{t_{j+1}} g_\beta(t)dt$;

Let $j = j + 1$;

End Do

Figure 5.5: The implemented parallel algorithm of the Piecewise Adjoint method - continued

Evaluate *vectors* $\{s_1, \ldots, s_{N_j}\}$, *where* $t_{N_j+1} \equiv t^2$ *is the final time, by solving the linear system of Eq.* (5.78);

Assemble $I^{\mu,\nu} = \tau^T + \sum_{j=1}^{N_j} {s_j}^\top T_j$;

Evaluate $\mu(t^1), \mu'(t^1)$ *of Eqs.* (5.107) *and* (5.108);

Evaluate *the* **initial terms** $\mu^\top(t^1)k_1$ *and* ${\mu'}^\top(t^1)M(t^1)q_\beta(t^1)$;

Evaluate *the* **final terms** $l_\beta(t^2), {\gamma^2}^\top \Phi_\beta(t^2), {\eta^2}^\top(\Phi_q \hat{q}')_\beta(t^2)$, *and* $\xi^2 \Omega_\beta(t^2)$;

Evaluate *the gradient of the functional of Eq.* (5.103), *by assembling* $I_g, I^{\mu,\nu}$, the **initial terms**, *the* **final terms**, *and* g_1;

Destroy *the* **ChildrenThreads**;

Figure 5.6: The implemented parallel algorithm of the Piecewise Adjoint method - concluded

t^2. The following computing tasks are identified inside the main loop:

1. Implicit integration of the DAE of motion of Eq. (4.20).

2. Factorization of the constraint Jacobian Φ_q and of the non-singular block Φ_u for each point of the mesh $\Delta^j, j = 1, 2, \ldots, N_j$ defined by Eq. (5.72).

3. Evaluation of the columns of the fundamental matrices $\{{Y_0}^j(t_k), t_k \in \Delta^j\}$ and the particular solutions $\{v^j(t_k), t_k \in \Delta^j\}$ through the integrate-recover-assemble algorithm defined in Section 5.2.

4. Updating of τ of Eq. (5.136) and evaluation of T_j of Eq. (5.137).

Assume that the implicit integration of the DAE of motion of Eq. (4.20) using the LLNL's IDA integrator [31] require

Let $t_1 = t^1$; $j = 1$;

Evaluate IC

$q(\beta, t_1), q'(\beta, t_1), q''(\beta, t_1), \lambda(\beta, t_1)$;

$q_\beta(\beta, t_1), q'_\beta(\beta, t_1), q''_\beta(\beta, t_1), \lambda_\beta(\beta, t_1)$;

Evaluate $g_1 = g_{q'}(t^1) q_\beta(t^1)$; $\tau = 0$; $I_g = 0$;

Do while $|\Omega(t_j, q(\beta, t_j), \beta)| > \epsilon$

pipeline stage 1

define *subinterval* $\mathcal{I}^j = [t_j, t_{j+1}]$ *and* **fine-grid**

$\Delta^j = \{t_{k_1} = t_j, \ldots, t_{k_2} = t_{j+1}\}$;

solve *the DAE of motion of Eq.* (4.20) *on* Δ^j;

store $\{q(\beta, t_k), q'(\beta, t_k), q''(\beta, t_k), \lambda(\beta, t_k), t_k \in \Delta^j\}$;

independently *compute the LU factorization of* Φ_q *at each mode* $t_k \in \Delta^j$;

store *the factored constraint Jacobians* $\{\Phi_q^{LU}(t_k), t_k \in \Delta^j\}$;

End pipeline stage 1

pipeline stage 2

For $i = 1, 2, \ldots, 2d + 1$

by **interpolating** *the set*

$\{q(\beta, t_k), q'(\beta, t_k), q''(\beta, t_k), \lambda(\beta, t_k), t_k \in \Delta^j\}$ *and using the set*

$\{\Phi_q^{LU}, t_k \in \Delta^j\}$,

independently *evaluate the columns of fundamental matrices*

$\{Y_0{}^j(t_k), t_k \in \Delta^j\}$ *and the particular solutions* $\{v^j(t_k), t_k \in \Delta^j\}$,

through the **integrate-recover-assemble** *algorithm, defined*

in Section 5.2;

End For

store $Y_0{}^j(t_j), v^j(t_j)$;

End pipeline stage 2

Figure 5.7: The pipelined algorithm for the evaluation of gradients of functionals through the Piecewise Adjoint method

pipeline stage 3

using $\{Y_0^j(t_k), v^j(t_k), t_k \in \Delta^j\}$ **update** τ *of Eq.* (5.136), T_j *of Eq.* (5.137);

update $I_g = I_g + \int_{t_j}^{t_{j+1}} g_\beta(t)dt$;

End pipeline stage 3

Let $j = j + 1$;

End Do

Evaluate *vectors* $\{s_1, \ldots, s_{N_j}\}$, *where* $t_{N_j+1} \equiv t^2$ *is the final time, by solving the linear system of Eq.* (5.78);

Assemble $I^{\mu,\nu} = \tau^T + \sum_{j=1}^{N_j} {s_j}^\top T_j$;

Evaluate $\mu(t^1), \mu'(t^1)$ *of Eqs.* (5.107) *and* (5.108);

Evaluate *the* **initial terms** $\mu^\top(t^1)k_1$ *and* ${\mu'}^\top(t^1)M(t^1)q_\beta(t^1)$;

Evaluate *the* **final terms** $l_\beta(t^2), {\gamma^2}^\top \Phi_\beta(t^2), {\eta^2}^\top(\Phi_q \hat{q}')_\beta(t^2)$, *and* $\xi^2 \Omega_\beta(t^2)$;

Evaluate *the gradient of the functional of Eq.* (5.103), *by assembling* $I_g, I^{\mu,\nu}$, the **initial terms**, *the* **final terms**, *and* g_1;

Figure 5.8: The pipelined algorithm for the evaluation of gradients of functionals through the Piecewise Adjoint method - continued

1. N_{steps}^{EOM} integration steps.

2. N_r^{EOM} evaluations of the $2(n+m)$-dimensional vector function $F(y, y', t)$ of Eq. (4.20).

3. N_J^{EOM} evaluations of the $(2(n+m)) \times (2(n+m))$ Jacobian J of Eq. (4.22).

4. One factorization per step of the Jacobian J of Eq. (4.22), due to the integrator's correction phase [31]. The factorization of a $s \times s$ matrix A requires $c \times s^3$ flops, where constant c depends on the factorization method [8]; e.g., $c = \frac{2}{3}$ for LU-factorization [8]. Hence, the factorization of the $(2(n+m)) \times (2(n+m))$ Jacobian J of Eq. (4.22) requires $64c(n+m)^3$ flops per integration step.

5. N_{Cf}^{EOM} integrator correction failures [31] that require re-factorization of the Jacobian J of Eq. (4.22).

Therefore, the total number of flops required by the integration of the DAE of motion of Eq. (4.20) is

$$\begin{aligned} N^{EOM} &= N_r^{EOM} \times 2(n+m) + N_J^{EOM} \times 4(n+m)^2 \\ &+ N_{steps}^{EOM} \times 64c(n+m)^3 + N_{Cf}^{EOM} \times 64c(n+m)^3 \\ &= 64c(n+m)^3(N_{steps}^{EOM} + N_{Cf}^{EOM}) \\ &+ 4N_J^{EOM}(n+m)^2 + 2N_r^{EOM}(n+m) \end{aligned} \tag{5.142}$$

Factorizations of the $m \times n$ constraint Jacobian Φ_q and of the $m \times m$ block Φ_u require $c \times m^3$ flops, each. These two factorizations are performed once per mesh point. Assuming the total simulation time interval $[t^1, t^2]$ contains N_{grid} mesh points,

the factorizations of the constraint Jacobian Φ_q and block Φ_u require a number of flops of

$$N^{FACTS} = N_{grid} \times (cm^3 + cm^3) = 2cN_{grid}m^3 \tag{5.143}$$

The integrate-recover-assemble algorithm defined in Section 5.2 is called $2d+1$ times to evaluate the $2d$ columns of the fundamental matrix $Y_0{}^j(t_k)$ and the particular solutions $v^j(t_k)$ at each mesh point $t_k \in \Delta^j$. Each time it is called, the integrate-recover-assemble algorithm integrates the adjoint DAE of Eq. (4.194) with different initial conditions and different right-sides. Assume that the implicit integration of the adjoint DAE of Eq. (4.20) using the LLNL's IDA integrator [31] require

1. N_{steps}^{ADAE} integration steps.

2. N_r^{ADAE} evaluations of the $2(n+m)$-dimensional vector function $F_a(y, y', t)$ of Eq. (4.194).

3. N_J^{ADAE} evaluations of the $(2(n+m)) \times (2(n+m))$ Jacobian J_a of Eq. (4.196).

4. One factorization per step of the Jacobian J_a of Eq. (4.196), due to the integrator's correction phase [31]. The factorization of the $(2(n+m)) \times (2(n+m))$ Jacobian J_a of Eq. (4.196) requires $64c(n+m)^3$ flops per integration step.

5. N_{Cf}^{ADAE} integrator correction failures [31] that require re-factorization of the Jacobian J_a of Eq. (4.196).

Therefore, the total number of flops required by the integration of one Adjoint DAE is

$$\begin{aligned} N^{ADAE} &= N_r^{ADAE} \times 2(n+m) + N_J^{ADAE} \times 4(n+m)^2 \\ &+ N_{steps}^{ADAE} \times 64c(n+m)^3 + N_{Cf}^{ADAE} \times 64c(n+m)^3 \\ &= 64c(n+m)^3(N_{steps}^{ADAE} + N_{Cf}^{ADAE}) \\ &+ 4N_J^{ADAE}(n+m)^2 + 2N_r^{ADAE}(n+m) \end{aligned} \tag{5.144}$$

Since the integrate-recover-assemble algorithm is called $2d+1$ times to evaluate the fundamental matrices $\{Y_0{}^j(t_k), t_k \in \Delta^j\}$ and the particular solutions $\{v^j(t_k), t_k \in \Delta^j\}$, its number of flops is on average $(2d+1)N^{ADAE}$.

Vector $\tau_{j,0}^{\mu}$ of Eq. (5.120) is evaluated by applying a quadrature numerical formula; e.g., Simpson method [8] to the product of n-dimensional row vector a_j^{μ} of Eq. (5.95) and the $n \times n_\beta$ matrix K_2 of Eq. (5.105) on the mesh Δ^j. Each such product requires $n \times n_\beta$ multiplications. The quadrature numerical formula requires the evaluation of this product for each of the N_{grid} mesh points. Hence, the number of flops required by the evaluation of vector $\tau_{j,0}^{\mu}$ is

$$N(\tau_{j,0}^{\mu}) = N_{grid} \times n \times n_\beta \tag{5.145}$$

Matrix T_j^{μ} of Eq. (5.121) is evaluated by applying a quadrature numerical formula to the product of the transposed of the $n \times 2d$ matrix B_j^{μ} of Eq. (5.96) and the $n \times n_\beta$ matrix K_2 of Eq. (5.105) on the mesh Δ^j. Each such product requires $2d \times n \times n_\beta$ multiplications. The quadrature numerical formula requires the evaluation of this product for each of the N_{grid} mesh points. Hence, the number of flops required

by the evaluation of matrix T_j^μ is

$$N(T_j^\mu) = N_{grid} \times 2d \times n \times n_\beta \tag{5.146}$$

Vector $\tau_{j,0}^\nu$ of Eq. (5.129) is evaluated by applying a quadrature numerical formula to the product of m-dimensional row vector a_j^ν of Eq. (5.98) and the $m \times n_\beta$ matrix Φ_β on the mesh Δ^j. Each such product requires $m \times n_\beta$ multiplications. The quadrature numerical formula requires the evaluation of this product for each of the N_{grid} mesh points. Hence, the number of flops required by the evaluation of vector $\tau_{j,0}^\nu$ is

$$N(\tau_{j,0}^\nu) = N_{grid} \times m \times n_\beta \tag{5.147}$$

Matrix T_j^ν of Eq. (5.130) is evaluated by applying a quadrature numerical formula to the product of the transposed of the $m \times 2d$ matrix B_j^ν of Eq. (5.99) and the $m \times n_\beta$ matrix Φ_β on the mesh Δ^j. Each such product requires $2d \times m \times n_\beta$ multiplications. The quadrature numerical formula requires the evaluation of this product for each of the N_{grid} mesh points. Hence, the number of flops required by the evaluation of matrix T_j^ν is

$$N(T_j^\nu) = N_{grid} \times 2d \times m \times n_\beta \tag{5.148}$$

In addition, vector a_j^ν of Eq. (5.98), which is used to compute the vector $\tau_{j,0}^\nu$ of Eq. (5.129), requires the evaluation of the inverse of $(n+m) \times (n+m)$ matrix M^S of Eq. (5.89) for each mesh point. Also, matrix B_j^ν of Eq. (5.99), which is used to compute the matrix T_j^ν of Eq. (5.130), requires $M^{S^{-1}}$ for each mesh point. Therefore, computing $\tau_{j,0}^\nu$ and T_j^ν requires the factorization of matrix (M^S), for each of the N_{grid}

mesh points; i.e., the number of flops required is

$$N^{\nu}_{FACT} = N_{grid} \times c(n+m)^3 = cN_{grid}(n+m)^3 \tag{5.149}$$

Vector τ of Eq. (5.136) and matrix T_j of Eq. (5.137) are updated by evaluating vectors $\tau^{\mu}_{j,0}$ and $\tau^{\nu}_{j,0}$ and matrices T^{μ}_j and T^{ν}_j. Hence, the number of flops required for the evaluation of τ and T_j is

$$\begin{aligned} N^{T_j,\tau} &= N(\tau^{\mu}_{j,0}) + N(T^{\mu}_j) + N(\tau^{\nu}_{j,0}) + N(T^{\nu}_j) + N^{\nu}_{FACT} \\ &= N_{grid}(2d+1)(n+m)n_{\beta} + cN_{grid}(n+m)^3 \end{aligned} \tag{5.150}$$

As a result, the total number of flops for the Piecewise Adjoint method is

$$\begin{aligned} N_{PA}(n,m,n_{\beta}) &= N^{EOM} + N^{FACTS} + (2d+1)N^{ADAE} + N^{T_j,\tau} \\ &= (n+m)^3\big(64c(N^{EOM}_{steps} + N^{EOM}_{Cf} + (2d+1)(N^{ADAE}_{steps} + N^{ADAE}_{Cf})) + cN_{grid}\big) \\ &+ (n+m)^2\big(4N^{EOM}_{J} + 4(2d+1)N^{ADAE}_{J}\big) \\ &+ (n+m)\big(2N^{EOM}_{r} + 2(2d+1)N^{ADAE}_{r}\big) \\ &+ 2cN_{grid}m^3 + N_{grid}(2d+1)(n+m)n_{\beta} \end{aligned} \tag{5.151}$$

The execution time of the sequential Piecewise Adjoint method is proportional to the number N_{PA} of floating-point operations [29]

$$\tau^{seq}_{PA} = \alpha N_{PA}(n,m,n_{\beta}) \tag{5.152}$$

where α is a constant depending on the CPU type, memory access speed, and operating system [54].

5.5.2 Multiprocessor Estimates

Assume now that the following shared memory parallel architecture with kernel threads [54] is used: the number of available parallel processors is n_p, there are n_{th} children threads in addition to the main thread of a process [54], and that POSIX threads [15] are used. Multi-threading structures; e.g., *mutexes* and *condition variables* are used to prevent threads from writing into the same memory location at the same time and for communication between threads [15]. Also, the main thread requires barriers [15]. Barriers allow the main thread to wait for the other threads to finish and then collect their results; e.g., a producer-consumer application [54] in which the main thread produces some data, the children threads process it, and the main thread waits to collect the results. With POSIX threads barriers are implemented using a pair of mutexes and a condition variable [15]. The use of mutexes and condition variables is usually computationally expensive and should be kept to a minimum [15]. The time that the operating system spends on communication between threads through mutexes and condition variables is called synchronization time and it depends on the computer architecture used, number of processors, and number of threads.

If a computing task X can be split into n_{th} threads of computation and the number of floating point operations associated with X, performed sequentially, is N_X^{seq} then the number of floating point operations per thread of computing task X is

$$N_X^{n_{th}} = \frac{N_X^{seq}}{n_{th}} \tag{5.153}$$

If $S_X(n_{th}, n_p)$ is the synchronization time of the n_{th} threads on n_p processors for task X, then the estimated execution time of task X when the operating system is responsible for assigning the n_{th} threads to the n_p available processors is [15, 19, 29]

$$\tau_X(n_{th}, n_p) = \alpha \frac{N_X^{seq}}{n_p} + S_X(n_{th}, n_p) \tag{5.154}$$

where α is a constant depending on the CPU type, memory access speed, and operating system [54]. When the number of available processors n_p is less than the number of necessary threads n_{th}, there is the alternate option of partitioning the n_{th} threads into $\frac{n_{th}}{n_p}$ groups of n_p threads, execute all threads in a group in parallel, but execute the groups one after another. This option is in general more expensive [19] because the number of barriers is increased by a factor of $\frac{n_{th}}{n_p}$. Therefore, this option is not implemented.

For the implemented parallel Piecewise Adjoint algorithm of Figs. 5.4 through 5.6 each of the following computing tasks are performed in parallel: (1) factorization of the Φ_q and Φ_u in each point of the Δ^j mesh and (2) evaluation of the columns of the fundamental matrices ${Y_0}^j(t)$ and the particular solutions $v^j(t)$ through the integrate-recover-assemble algorithm. Substituting N^{FACTS} of Eq. (5.143) for N_X^{seq} into Eq. (5.154), it follows that the estimated execution time for the factorization of Φ_q and Φ_u, on the parallel architecture with n_p processors and n_{th} threads, is

$$\tau_{FACTS}(n_{th}, n_p) = \alpha \frac{2cN_{grid}m^3}{n_p} + S_{FACTS}(n_{th}, n_p) \tag{5.155}$$

Also, substituting the $(2d+1)N^{ADAE}$ flops required for the evaluation of the columns of the fundamental matrices and the particular solutions, where N^{ADAE} is defined by

Eq. (5.144), for N_X^{seq} into Eq. (5.154), it follows that the evaluation of the columns of the fundamental matrices ${Y_0}^j(t)$ and the particular solutions $v^j(t)$ through the integrate-recover-assemble algorithm, on the parallel architecture with n_p processors and n_{th} threads, is

$$\begin{aligned}\tau_{ADAE\times(2d+1)}(n_{th},n_p) &= \alpha\frac{64c(2d+1)(n+m)^3(N_{steps}^{ADAE}+N_{Cf}^{ADAE})}{n_p}\\ &+ \alpha\frac{4(2d+1)N_J^{ADAE}(n+m)^2+2(2d+1)N_r^{ADAE}(n+m)}{n_p}\\ &+ S_{ADAE\times(2d+1)}(n_{th},n_p)\end{aligned} \tag{5.156}$$

The integration of the DAE of motion and the evaluation of τ of Eq. (5.136) and T_j of Eq. (5.137) are performed sequentially. Therefore,

$$\tau_{EOM}(n_{th},n_p)=\alpha N^{EOM} \tag{5.157}$$

where N^{EOM} is defined by Eq. (5.142) and

$$\tau_{T_j,\tau}(n_{th},n_p)=\alpha N^{T_j,\tau} \tag{5.158}$$

where $N^{T_j,\tau}$ is defined by Eq. (5.150).

As a result, the estimated execution time for the implemented parallel Piecewise Adjoint algorithm of Figs. 5.4 through 5.6 on the parallel architecture with n_p processors and n_{th} threads is

$$\begin{aligned}\tau_{PA}^{parallel}(n_{th},n_p) &= \tau_{EOM}(n_{th},n_p)+\tau_{T_j,\tau}(n_{th},n_p)\\ &+ \tau_{ADAE\times(2d+1)}(n_{th},n_p)+\tau_{FACTS}(n_{th},n_p)\end{aligned} \tag{5.159}$$

where $\tau_{EOM}(n_{th},n_p)$ is defined by Eq. (5.157), $\tau_{T_j,\tau}(n_{th},n_p)$ is defined by Eq. (5.158), $\tau_{ADAE\times(2d+1)}(n_{th},n_p)$ is defined by Eq. (5.156), and $\tau_{FACTS}(n_{th},n_p)$ is defined by Eq.

(5.155). The number of necessary threads, in addition to the process' main thread, for the factorization of Φ_q and Φ_u in each point of the Δ^j mesh is the average number $|\Delta^j|_{j=1,2,\ldots,N_j}$ of mesh points in the set of meshes $\{\Delta^j, j = 1, 2, \ldots, N_j\}$. The number of necessary threads, in addition to the process' main thread, for the evaluation of the columns of the fundamental matrices ${Y_0}^j(t)$ and the particular solutions $v^j(t)$, through the integrate-recover-assemble algorithm, is $2d+1$. Hence, the total number of necessary threads for the implemented parallel Piecewise Adjoint algorithm is

$$n_{th}^{PA} = max\{|\Delta^j|_{j=1,2,\ldots,N_j} + 1, 2d + 2\} \tag{5.160}$$

In order to minimize the amount of time spent on synchronization the number of processors must be at least as big as the number of threads [19]

$$n_p \geq n_{th}^{PA}$$

Consider now the pipelined Piecewise Adjoint algorithm of Figs. 5.7 and 5.8. In the first pipeline stage the computing tasks that are executed are the integration of the DAE of motion followed by the factorizations of Φ_q and Φ_u. Hence, the estimated execution time for the first pipeline stage is

$$\tau_{PA}^{p_1} = \tau_{EOM}(n_{th}, n_p) + \tau_{FACTS}(n_{th}, n_p) \tag{5.161}$$

where $\tau_{EOM}(n_{th}, n_p)$ is defined by Eq. (5.157) and $\tau_{FACTS}(n_{th}, n_p)$ is defined by Eq. (5.155). The number of necessary threads for the first pipeline stage is the average number $|\Delta^j|_{j=1,2,\ldots,N_j}$ of mesh points in the set of meshes $\{\Delta^j, j = 1, 2, \ldots, N_j\}$.

In the second pipeline stage the computing tasks that are executed are the evaluations of the columns of the fundamental matrices ${Y_0}^j(t)$ and the particular so-

lutions $v^j(t)$ through the integrate-recover-assemble algorithm. Hence, the estimated execution time for the second pipeline stage is

$$\tau_{PA}^{p_2} = \tau_{ADAE\times(2d+1)}(n_{th}, n_p) \tag{5.162}$$

where $\tau_{ADAE\times(2d+1)}(n_{th}, n_p)$ is defined by Eq. (5.156). The number of necessary threads for the second pipeline stage is $2d + 1$.

In the third pipeline stage the computing task that is executed is the evaluation of vector τ of Eq. (5.136) and matrix T_j of Eq. (5.137). Hence, the estimated execution time for the third pipeline stage is

$$\tau_{PA}^{p_3} = \tau_{T_j,\tau}(n_{th}, n_p) \tag{5.163}$$

where $\tau_{T_j,\tau}(n_{th}, n_p)$ is defined by Eq. (5.158). The number of necessary threads for the third pipeline stage is 1.

Since the three pipeline stages are executed simultaneously, the estimated execution time for the pipelined Piecewise Adjoint algorithm is the maximum execution time of the three pipeline stages

$$\tau_{PA}^{pipelined} = max\{\tau_{PA}^{p_1}, \tau_{PA}^{p_2}, \tau_{PA}^{p_3}\} \tag{5.164}$$

and the number of necessary threads, in addition to the process' main thread, is the sum of necessary threads for each pipeline stage

$$n_{th}^{pipelined} = |\Delta^j|_{j=1,2,\ldots,N_j} + 2d + 3 \tag{5.165}$$

The number of flops $N_{PA}(n, m, n_\beta)$ of Eq. (5.151) for the Piecewise Adjoint method increases with the number of design parameters n_β. Therefore, the execution

times of both the sequential and the parallel Piecewise Adjoint algorithms, of Eqs. (5.152) and (5.159), respectively, increases with the number of design parameters n_β.

The execution time for the implemented parallel Piecewise Adjoint algorithm is difficult to estimate in advance because the time spent by the operating system on synchronizing mutexes and condition variables [15] is difficult to measure. Although the estimation of Eq. (5.159) includes synchronization costs, they are difficult to determine in actual parallel experiments [15].

5.6 Memory Load Analysis

In the Adjoint method, storage of the equations of motion states, q, v, a, and λ is required during forward integration of the DAE of motion, to be used for assembling and solving the Adjoint DAE using backward integration. In the Piecewise Adjoint method, at each time t_j, $j = 1, 2, \ldots, N_j$; i.e., the beginning of the fine-grid interval $\mathcal{I}^j$ and also the end point of the backward integration of the Adjoint CPUODE on fine-grid interval $\mathcal{I}^j$, the fundamental matrix $Y_0^j(t_j)$ and the particular solution $v^j(t_j)$ need to be stored, in order to construct and solve the linear system of Eq. (5.78), after final time $t^2 \equiv t_{N_j+1}$ is reached. Also, matrix T_j defined by Eq. (5.114) and vector $\tau = \sum_{j=1}^{N_j} \tau_j$, which accumulates the sum of vectors τ_j defined by Eq. (5.113) for each fine-grid interval $\mathcal{I}^j$, need to be stored in order to assemble the gradient of the functional after final time is reached, as shown by the algorithm in Figs. 5.7 and 5.8.

Consider a multibody system with n_b bodies, m constraints (including the

Euler parameter normalization constraints), and n_β design parameters. Also, consider N_j fine-grid intervals $\mathcal{I}^j$, $j = 1, 2, \ldots, N_j$. Each fine-grid interval consists of ρ mesh points. For the Adjoint method, when final time is reached $N_j \times \rho$ sets of motion states q, v, a, and λ are stored, resulting in a total of

$$N^A = (3n + m) \times N_j \times \rho \tag{5.166}$$

double precision numbers, where $n = 7 \times n_b$.

For the Piecewise Adjoint method, the dimension of fundamental matrices Y_0^j is $2d \times 2d$, where $d = n - m$; the dimension of vectors v^j is $2d$; the size of vector τ is n_β; and the dimension of matrices T_j is $n_\beta \times 2d$. Therefore, when final time is reached a total of

$$N^{PA} = (n_\beta(2d + 1) + 4d^2 + 4d) \times N_j \tag{5.167}$$

double precision numbers are stored.

For the slider-crank model, with $n = 28$, $m = 27$, and $d = 1$; $n_\beta = 10$ design parameters; and $N_j = 40$ fine-grid intervals, each with $\rho = 10$ mesh points, the Adjoint method requires storage of $N^A = 44,400$ double precision points, while the Piecewise Adjoint method requires storage of $N^{PA} = 1520$ double precision points, resulting in less than 3.5% of the Adjoint method memory consumption. For $n_\beta = 100$ design parameters, $N^{PA} =$ which is less than a third of the Adjoint method memory consumption.

For the HMMWV 13 model, with $n = 91$, $m = 81$, and $d = 10$; $n_\beta = 10$ design parameters; and $N_j = 40$ fine-grid intervals, each with $\rho = 10$ mesh points,

the Adjoint method requires storage of $N^A = 141,600$ double precision points, while the Piecewise Adjoint method requires storage of $N^{PA} = 26,000$ double precision points, resulting in less than a fifth of the Adjoint method memory consumption. For $n_\beta = 100$ design parameters, $N^{PA} = 57600$ which is less than half of the Adjoint method memory consumption.

CHAPTER 6
NUMERICAL IMPLEMENTATION AND PRELIMINARY RESULTS

This Chapter is organized as follows: Section 6.1 shows error control of the quadrature methods that are used; Section 6.2 presents the slider-crank and the vehicle models; Sections 6.3 and 6.4 show validation results of the implementation of the Direct Differentiation, Adjoint, and Piecewise Adjoint methods; and Section 6.5 presents (1) parallel implementation attempts on the two parallel architectures, (2) an analysis of the sequential and parallel Piecewise Adjoint methods, and (3) predictions of the pipelined parallel Piecewise Adjoint method on multiprocessors architectures with adequate number of processors. Numerical results are presented for two multibody systems, a slider-crank and a vehicle. For each model, the following equations are solved:

1. The index-1 DAE of motion of Eq. (4.20), presented in Section 4.2.

2. The index-1 Direct Differentiation DAE of Eq. (4.113), presented in Section 4.3.

3. The index-1 Adjoint DAE of Eq. (4.194), presented in Section 4.4.

4. The non-homogeneous and homogeneous Adjoint CPUODE of Eq. (5.31), using the integrate-recover-assemble algorithm presented in Section 5.2.

The solutions of these equations are used to integrate gradients of functionals of the form

$$\Psi = \int_{t^1}^{t^2} (\alpha_1 q^\top q + \alpha_2 q'^\top q')dt \tag{6.1}$$

where α_1 and α_2 are constant scalars. Sequential and parallel experiments are performed on computers with the following configurations of microprocessor type (CPU), microprocessor speed, random access memory (RAM), and operating system (OS):

1. single-processor computer with CPU: AMD Athlon XP 2500+, speed: 1.8 GHz, RAM: 512 MB RAM, and OS: Suse Linux 9.0;

2. dual-processor computer with 2× CPU: Intel Xeon , speed: 3 GHz, RAM: 1GB RAM, and OS: Windows XP;

3. quad-processor computer with 4× CPU: AMD Opteron Processor 850, speed: 2.4 GHz, RAM: 21 GB, and OS: Suse Linux 9.1;

6.1 Error Control

The absolute and relative tolerances for the integration of the DAE of motion of Eq. (4.20); the DAE of Eq. (4.113) of the Direct Differentiation method, presented in Section 4.3; and the DAE of Eq. (4.194) of the Adjoint and the Piecewise Adjoint methods, of Sections 4.4 and 5.2, respectively, are controlled by the IDA DAE solver [31] through a variable step-size and variable order *constant leading coefficient* algorithm [13]. For the control of the error in the integration of the functional and its

gradient through the Direct Differentiation, Adjoint, and Piecewise Adjoint methods, two options are implemented:

1. Error control by using Richardson extrapolation [25];

2. Error control by using the IDA integrator;

Let $I^f_{[t_1,t_2]}$ be a numerical integration method applied to function $f(t)$ on interval $[t_1, t_2]$; e.g., Simpson method [8]. Using Richardson extrapolation, an estimate of the local error of the integration is obtained by evaluating the difference between integrating with step-size $h = t_2 - t_1$, and integrating with half of the step-size h; i.e.,

$$e_{[t_1,t_2]} = I^f_{[t_1,t_2]} - (I^f_{[t_1,\frac{t_1+t_2}{2}]} + I^f_{[\frac{t_1+t_2}{2},t_2]})$$

If $e_{[t_1,t_2]}$ is less than a prescribed *error-per-length*, obtained by dividing the tolerance by the length of the interval, $(t_2 - t_1)$, then $I^f_{[t_1,t_2]}$ is accepted; otherwise, $I^f_{[t_1,t_2]}$ is rejected, the interval is divided in half, and the result is obtained by recursively applying the same algorithm to each sub-interval, and add results on sub-intervals.

For the second method of error control in the integration of function $f(t)$ on interval $[t_1, t_2]$, consider

$$z(t) = \int_{t_1}^{t} f(\tau) d\tau$$

Since $I = z(t_2)$, the integration of function $f(t)$ on interval $[t_1, t_2]$ is performed by solving the ODE

$$z'(t) = f(t)$$

with initial conditions $z(t_1) = 0$. Using the IDA solver, the local error is controlled by the IDA integrator. Tests with both methods resulted in the same order of accuracy,

although the IDA solver, which uses an implicit integration method, is less efficient in terms of execution speed.

6.2 Slider-Crank and Vehicle Models

In this Section the topology graphs [52] of the spatial slider-crank and the thirteen-body High mobility multipurpose wheeled vehicle (HMMWV) are presented. The nodes of the graphs represent bodies and the edges represent joints in the set $\{D, S, R, U, T\}$, where labels D, S, R, U, and T represent distance, spherical, revolute, universal, and translational joints, respectively.

Figure 6.1 presents the spatial slider-crank. It consists of the following four bodies:

1. ground;
2. crank;
3. connecting rod (CONROD);
4. slider;

The bodies are connected through a set of four joints. Table 6.1 presents the set of joints of the spatial slider-crank, where each entry presents the type of the joint and the bodies it connects.

	Type	Body i	Body j
1	Spherical	CRANK	CONROD
2	Universal	CONROD	SLIDER
3	Translational	SLIDER	GROUND
4	Revolute	GROUND	CRANK

Table 6.1: The joints of the spatial slider-crank.

Figure 6.2 presents the spatial slider-crank topology graph.

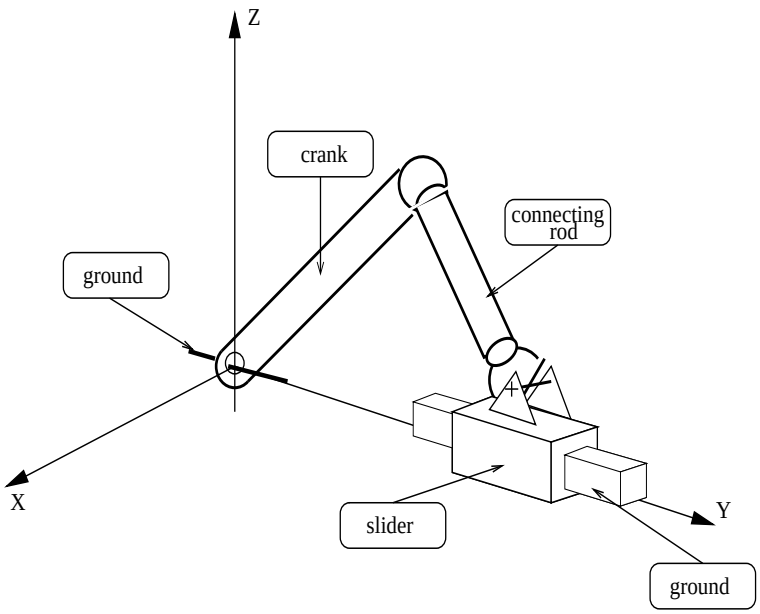

Figure 6.1: The spatial slider-crank

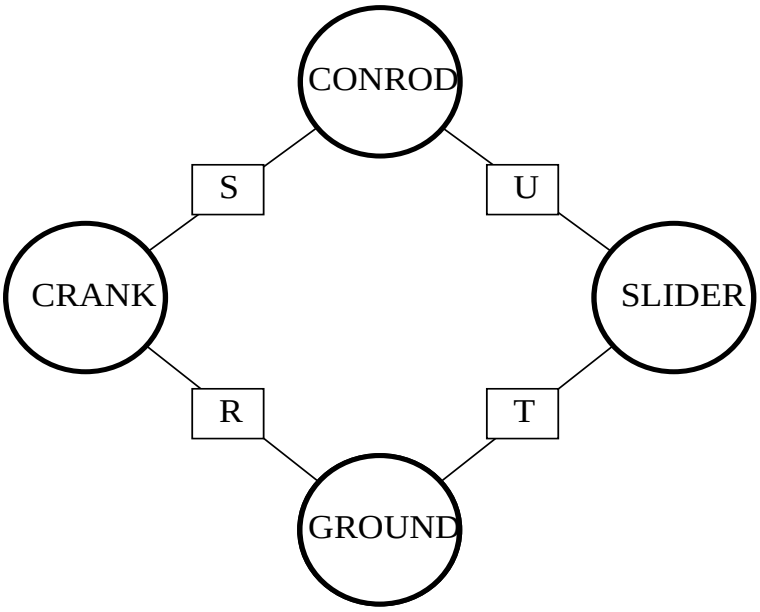

Figure 6.2: The slider-crank topology

The thirteen bodies HMMWV model (HMMWV 13) is similar to the fourteen bodies HMMWV model (HMMWV 14) presented by Negrut [41]. Tires are modeled as vertical translational-spring-damper elements between bodies and the terrain, which is modeled as a planar surface. In order to fix the steering rack, the HMMWV 14 model [41] has a translational and a distance joint between the chassis and the steering rack. In the HMMWV 13 model, the steering rack is removed and its mass and inertia are transfered to the chassis. In both HMMWV models, stiffness is introduced by prescribing very large values for the stiffness and damping coefficients of the Tire and TSDA force elements; i.e., the stiffness coefficient of each TSDA is $2.0e+7$ N/m and the damping coefficient is $2.0e+6$ Ns/m. The stiffness coefficients of the tires are 296325 N/m and the damping coefficients are 3502 Ns/m. Figure 6.3

presents the HMMWV 13 model. It consists of the following thirteen bodies:

1. chassis;
2. front left lower control arm (FLLCA);
3. front left upper control arm (FLUCA);
4. front left wheel assembly (FLWA);
5. front right lower control arm (FRLCA);
6. front right upper control arm (FRUCA);
7. front right wheel assembly (FRWA);
8. rear left lower control arm (RLLCA);
9. rear left upper control arm (RLUCA);
10. rear left wheel assembly (RLWA);
11. rear right lower control arm (RRLCA);
12. rear right upper control arm (RRUCA);
13. rear right wheel assembly (RRWA);

Table 6.2 presents the set of joints of the HMMWV 13, where each entry presents the type of the joint and the bodies it connects.

	Type	Body i	Body j		Type	Body i	Body j
1	Distance	CHASSIS	FLWA	11	Distance	CHASSIS	RRWA
2	Revolute	CHASSIS	FLUCA	12	Revolute	CHASSIS	RRUCA
3	Revolute	CHASSIS	FLLCA	13	Revolute	CHASSIS	RRLCA
4	Spherical	FLWA	FLUCA	14	Spherical	RRWA	RRUCA
5	Spherical	FLWA	FLLCA	15	Spherical	RRWA	RRLCA
6	Distance	CHASSIS	FRWA	16	Distance	CHASSIS	RLWA
7	Revolute	CHASSIS	FRUCA	17	Revolute	CHASSIS	RLUCA
8	Revolute	CHASSIS	FRLCA	18	Revolute	CHASSIS	RLLCA
9	Spherical	FRWA	FRUCA	19	Spherical	RLWA	RLUCA
10	Spherical	FRWA	FRLCA	20	Spherical	RLWA	RLLCA

Table 6.2: The joints of the HMMWV 13.

Figure 6.4 presents the HMMWV 13 model topology graph.

Figure 6.3: The HMMWV

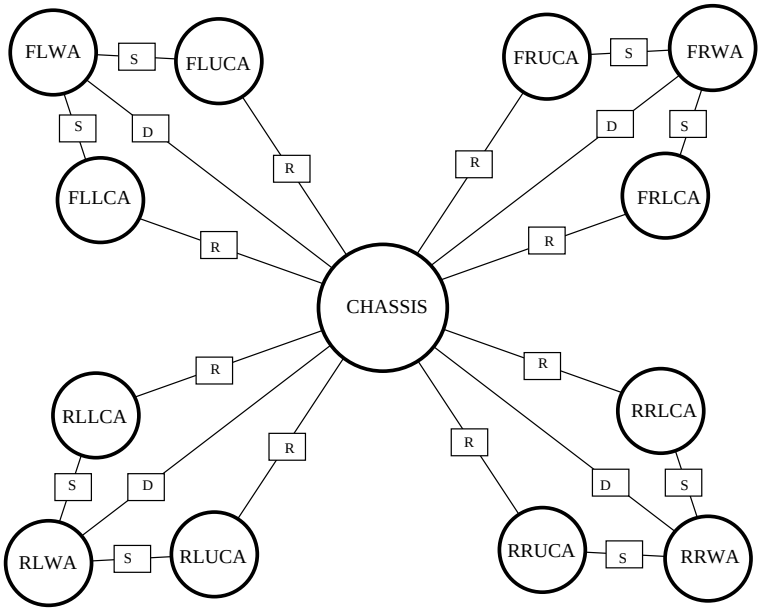

Figure 6.4: The HMMWV 13 topology

6.3 Validation of Kinematic and Force Derivatives

Analytic evaluation of kinematic and force derivatives has been validated by comparison with centered [8] finite differences,

$$\frac{\partial f(x)}{\partial x} = \frac{f(x+h) - f(x-h)}{2h} + O(h^2) \tag{6.2}$$

where $h = 10^{-6}$. Tables 6.3 and 6.4 show the absolute error between kinematic derivatives evaluated analytically and corresponding derivatives evaluated with finite differences, where the ^{FD} superscript stands for finite differences. The validation tests have been performed on the single-processor AMD Athlon computer. The vector norm used is the l_1 norm [9]

$$\|x\|_1 \equiv \sum_{i}^{n_x} |x_i|$$

where x is a n_x-dimensional vector.

	$\left\|\Phi_q^{FD} - \Phi_q\right\|_1$	$\left\|(\Phi_q\gamma)_q^{FD} - (\Phi_q\gamma)_q\right\|_1$
Distance	1.75e-09	1.21e-07
Spherical	9.77e-11	7.67e-09
Revolute	9.77e-11	7.67e-09
Universal	2.08e-10	6.45e-09
Translational	2.30e-10	1.77e-08

Table 6.3: Kinematic derivatives: Φ_q and $\Theta(\Phi, \gamma)$

	$\left\|((\Phi_q\gamma_1)_q\gamma_2)_q^{FD} - ((\Phi_q\gamma_1)_q\gamma_2)_q\right\|_1$	$\left\|({\Phi_q}^\top\zeta)_q^{FD} - ({\Phi_q}^\top\zeta)_q\right\|_1$
Distance	2.42e-06	4.51e-09
Spherical	0.0	7.36e-10
Revolute	1.80e-07	2.55e-09
Universal	7.86e-08	2.27e-09
Translational	3.66e-07	6.67e-09

Table 6.4: Kinematic derivatives: $({\Phi_q}^\top\zeta)_q$ and $\Upsilon(\Phi, \gamma_1, \gamma_2)$

Table 6.5 presents maximum error between analytic and finite difference evaluation of derivatives with respect to model parameters, Φ_β, $(\Phi_q\gamma)_\beta$, $((\Phi_q\gamma_1)_q\gamma_2)_\beta$, and

$(\Phi_q{}^\top\gamma)_\beta$. The vector β of model parameters depends on the type of joint for which derivatives are evaluated. For Distance and Spherical joints

$$\beta = \begin{pmatrix} s'^P_i{}^\top & s'^P_j{}^\top \end{pmatrix}^\top$$

for Revolute and Universal joints

$$\beta = \begin{pmatrix} s'^P_i{}^\top & s'^P_j{}^\top & h'_i{}^\top & h'_j{}^\top \end{pmatrix}^\top$$

and for the Translational joint

$$\beta = \begin{pmatrix} s'^P_i{}^\top & s'^P_j{}^\top & h'_i{}^\top & h'_j{}^\top & f'_i{}^\top & g'_i{}^\top \end{pmatrix}^\top$$

	$\max\{\|\Phi^{FD}_\beta - \Phi_\beta\|_1, \|(\Phi_q\gamma)^{FD}_\beta - (\Phi_q\gamma)_\beta\|_1,$ $\|((\Phi_q\gamma_1)_q\gamma_2)^{FD}_\beta - ((\Phi_q\gamma_1)_q\gamma_2)_\beta\|_1, \|(\Phi_q{}^\top\gamma)^{FD}_\beta - (\Phi_q{}^\top\gamma)_\beta\|_1\}$
Distance	3.41e-06
Spherical	8.74e-08
Revolute	8.74e-08
Universal	1.23e-07
Translational	6.30e-07

Table 6.5: Joint derivatives with respect to model parameters

Tables 6.6 and 6.7 present errors between analytic and finite difference evaluation of derivatives of forces with respect to generalized coordinates q and q' and

model parameters β. The vectorβ of model parameters depends on the type of force for which derivatives are evaluated. For constant applied forces

$$\beta = \begin{pmatrix} F_i^{A\top} & s_i'^{C\top} \end{pmatrix}^\top$$

for TSDA forces

$$\beta = \begin{pmatrix} s_i'^{P\top} & s_j'^{P\top} & k & c & l_0 & F \end{pmatrix}^\top$$

for Tire forces

$$\beta = \begin{pmatrix} s_i'^{P\top} & k & c & l_0 \end{pmatrix}^\top$$

for RSDA forces

$$\beta = \begin{pmatrix} f_i'^{\top} & f_j'^{\top} & g_i'^{\top} & h_i'^{\top} & h_j'^{\top} & k & c & N \end{pmatrix}^\top$$

and for Coriolis forces, L_i, $i = 1, 2, \ldots, n_b$,

$$\beta = \begin{pmatrix} m_i & s_i'^{C\top} & (J_i')_{11} & (J_i')_{12} & (J_i')_{13} & (J_i')_{22} & (J_i')_{23} & (J_i')_{33} \end{pmatrix}^\top$$

	$\|Q_q^{FD} - Q_q\|_1$	$\|Q_{q'}^{FD} - Q_{q'}\|_1$	$\|Q_\beta^{FD} - Q_\beta\|_1$
Constant	2.34e-09	0.00	2.00e-06
Tire	9.10e-09	8.85e-09	1.00e-08
TSDA	2.81e-08	3.63e-08	2.74e-06
RSDA	5.69e-10	1.22e-10	1.15e-05
Coriolis	9.95e-09	9.62e-09	1.58e-08

Table 6.6: Force derivatives

	$\left\|(Q_{q'}\gamma)_q^{FD} - (Q_{q'}\gamma)_q\right\|_1$	$\left\|(Q_{q'}\gamma)_{q'}^{FD} - (Q_{q'}\gamma)_{q'}\right\|_1$
Constant	0.00	0.00
Tire	2.34e-08	0.00
TSDA	8.95e-07	0.00
RSDA	2.08e-09	0.00
Coriolis	8.28e-08	6.05e-08

Table 6.7: Force derivatives

Table 6.8 presents errors between analytic and finite difference evaluation of derivatives involving the mass matrix with respect to generalized coordinates q and model parameters β. The vector of model parameters for the mass matrix is

$$\beta = \begin{pmatrix} m_i & s_i'^{C\top} & (J_i')_{11} & (J_i')_{12} & (J_i')_{13} & (J_i')_{22} & (J_i')_{23} & (J_i')_{33} \end{pmatrix}^\top$$

$\left\|(M\gamma)_q^{FD} - (M\gamma)_q\right\|_1$	$\left\|((M\gamma_1)_q\gamma_2)_q^{FD} - ((M\gamma_1)_q\gamma_2)_q\right\|_1$	$\left\|M_\beta^{FD} - M_\beta\right\|_1$
8.95e-09	1.37e-07	6.56e-09

Table 6.8: Mass matrix derivatives

The main source of the errors is the truncation error in the finite differences evaluation of derivatives [8]. There is no accumulation of round-off errors [8] in the

analytic evaluation of derivatives because derivatives are evaluated pointwise; i.e., their evaluation does not require summation over a time interval.

6.4 Validation of the Evaluation of Ψ_β Through Direct Differentiation, Adjoint, and Piecewise Adjoint Methods

In this Section the gradient of the functional of Eq. (6.1) is computed through the Direct Differentiation (DD), Adjoint (A), and Piecewise Adjoint (PA) methods and compared against the gradient of the same functional obtained with finite differences (FD). The numerical experiments have been performed on the single-processor AMD Athlon computer, the dual Intel Xeon computer, and the quad AMD Opteron computer. The time interval chosen is $[0, 2]$ sec, $\alpha_1 = \alpha_2 = 1$, and relative and absolute tolerances for the IDA solver are [31] of 10^{-5} and 10^{-6}, respectively. The tolerance used for integration of Ψ_β is 10^{-7}. The finite differences evaluation of Ψ_β uses the centered finite differences formula of Eq. (6.2) where $h = 10^{-6}$; i.e., the functional Ψ_{+h} is first integrated with each model parameter β_l, $l = 1, 2, \ldots, n_\beta$, perturbed to $\beta_l + h$, then Ψ_{-h} is integrated with each model parameter β_l perturbed to $\beta_l - h$, and

$$\Psi_\beta^{FD} = \frac{\Psi_{+h} - \Psi_{-h}}{2h} \tag{6.3}$$

Tables 6.9 through 6.11 present comparative results for the absolute errors between the Direct Differentiation, Adjoint, Piecewise Adjoint, and the finite differences meth-

ods for the slider-crank sensitivity with respect to sixteen model parameters. The validation results for the gradient of the functional are shown for each of the three computers because the different architectures and operating systems may influence the round-off errors [8]. The vector norm used is the l_1 norm. The spatial slider-crank sensitivity validation test case uses the following 16-dimensional vector of model parameters:

$$\beta^S = \left(\begin{array}{cccc} {s^{P_{rev}}}^\top & m_{CRANK} & (J'_{CONROD})_{i,j=1,2,3,i\neq j} & (J'_{SLIDER})_{i,j=1,2,3,i\neq j} \end{array} \right)^\top \tag{6.4}$$

where $s^{P_{rev}}$ is the 3-dimensional position vector of the revolute joint between ground and crank with respect to ground. Parameter m_{CRANK} is the crank mass. The six parameters $(J'_{CONROD})_{i,j=1,2,3,i\neq j}$ are the elements of the inertia tensor of the connecting rod. The six parameters $(J'_{SLIDER})_{i,j=1,2,3,i\neq j}$ are the elements of the inertia tensor of the slider.

	DD	A	PA
$\left\|\Psi_\beta^{FD} - \Psi_\beta\right\|_1$	5.03e-3	9.25e-3	7.12e-3

$\left\|\Psi_\beta^{Direct} - \Psi_\beta^{Piecewise}\right\|_1$	$\left\|\Psi_\beta^{Direct} - \Psi_\beta^{Adjoint}\right\|_1$
1.32e-2	9.86e-3

Table 6.9: The absolute error of Ψ_β with respect to FD, for the sensitivity of the slider-crank with sixteen model parameters on the AMD Athlon computer

	DD	A	PA
$\left\|\Psi_\beta^{FD} - \Psi_\beta\right\|_1$	1.27e-3	2.04e-3	1.8e-3
$\left\|\Psi_\beta^{Direct} - \Psi_\beta^{Piecewise}\right\|_1$		$\left\|\Psi_\beta^{Direct} - \Psi_\beta^{Adjoint}\right\|_1$	
5.76e-3		2.55e-3	

Table 6.10: The absolute error of Ψ_β with respect to FD, for the sensitivity of the slider-crank with sixteen model parameters on the Intel Xeon computer

	DD	A	PA
$\left\|\Psi_\beta^{FD} - \Psi_\beta\right\|_1$	8.13e-4	9.63e-4	9.98e-4
$\left\|\Psi_\beta^{Direct} - \Psi_\beta^{Piecewise}\right\|_1$		$\left\|\Psi_\beta^{Direct} - \Psi_\beta^{Adjoint}\right\|_1$	
1.39e-3		1.15e-3	

Table 6.11: The absolute error of Ψ_β with respect to FD, for the sensitivity of the slider-crank with sixteen model parameters on the AMD Opteron computer

Tables 6.12 through 6.17 present comparative results for the absolute errors between the Direct Differentiation, Adjoint, Piecewise Adjoint, and the finite differences methods for the HMMWV 13 sensitivity with respect to sixteen model parameters. The time interval chosen for the simulation is $[0, 2]s$ for the comparison between the Direct Differentiation, Adjoint, and finite differences methods. The time interval

chosen for the simulation is only $[0, 0.05]s$ for the comparison between the Piecewise Adjoint method and Direct Differentiation, Adjoint, and finite differences methods because the execution time of the Piecewise Adjoint method for the HMMWV 13 model is significantly larger, as will be explained in Section 6.5.2. The HMMWV 13 sensitivity validation test case uses the following 16-dimensional vector of model parameters:

$$\beta^H = \begin{pmatrix} {\beta^H}_{mass}^{\top} & (J'_{CHASSIS})_{i,j=1,2,3,i\neq j} & (J'_{RLLCA})_{i,j=1,2,3,i\neq j} \end{pmatrix}^{\top} \tag{6.5}$$

where vector

$${\beta^H}_{mass} = \begin{pmatrix} m_{CHASSIS} & m_{RLLCA} & m_{RLWA} & m_{RRWA} \end{pmatrix}^{\top}$$

consists of the masses of the chassis $m_{CHASSIS}$, rear left lower control arm m_{RLLCA}, rear left wheel assembly m_{RLWA}, and rear right wheel assembly m_{RRWA}. The six parameters $(J'_{CHASSIS})_{i,j=1,2,3,i\neq j}$ are the elements of the inertia tensor of the chassis. The six parameters $(J'_{RLLCA})_{i,j=1,2,3,i\neq j}$ are the elements of the inertia tensor of the rear left lower control arm.

	DD	A
$\left\|\Psi_\beta^{FD} - \Psi_\beta\right\|_1$	3.61e-03	7.25e-03
$\left\|\Psi_\beta^{Direct} - \Psi_\beta^{Adjoint}\right\|_1$		4.44e-04

Table 6.12: The absolute error of Ψ_β with respect to FD, for the sensitivity of the HMMWV 13 with sixteen model parameters on the AMD Athlon computer for the time interval of $[0, 2]s$

Method	DD	A	FD
$\|\Psi_\beta^{PA} - \Psi_\beta^{Method}\|_1$	1.05e-03	8.17e-04	9.65e-03

Table 6.13: The absolute error of Ψ_β with respect to PA, for the sensitivity of the HMMWV 13 with sixteen model parameters on the AMD Athlon computer for the time interval of $[0, 0.05]s$

	DD	A
$\|\Psi_\beta^{FD} - \Psi_\beta\|_1$	2.88e-03	6.61e-03
$\|\Psi_\beta^{Direct} - \Psi_\beta^{Adjoint}\|_1$		2.07e-04

Table 6.14: The absolute error of Ψ_β with respect to FD, for the sensitivity of the HMMWV 13 with sixteen model parameters on the Intel Xeon computer for the time interval of $[0, 2]s$

Method	DD	A	FD
$\|\Psi_\beta^{PA} - \Psi_\beta^{Method}\|_1$	8.77e-03	5.39e-04	7.1e-03

Table 6.15: The absolute error of Ψ_β with respect to PA, for the sensitivity of the HMMWV 13 with sixteen model parameters on the Intel Xeon computer for the time interval of $[0, 0.05]s$

	DD	A
$\left\|\Psi_\beta^{FD} - \Psi_\beta\right\|_1$	7.97e-04	6.56e-04
$\left\|\Psi_\beta^{Direct} - \Psi_\beta^{Adjoint}\right\|_1$		5.13e-04

Table 6.16: The absolute error of Ψ_β with respect to FD, for the sensitivity of the HMMWV 13 with sixteen model parameters on the AMD Athlon computer for the time interval of $[0, 2]s$

Method	DD	A	FD
$\left\|\Psi_\beta^{PA} - \Psi_\beta^{Method}\right\|_1$	5.11e-04	8.17e-04	1.82e-03

Table 6.17: The absolute error of Ψ_β with respect to PA, for the sensitivity of the HMMWV 13 with sixteen model parameters on the AMD Athlon computer for the time interval of $[0, 0.05]s$

In addition to the truncation error in the finite differences evaluation of the gradient of the functional, there are more significant round-off errors. There is accumulation of the round-off errors [8] in the evaluation of the gradient because it requires summation over a time interval.

6.5 Initial Assessment of the Parallel Implementation

Parallel experiments with the Piecewise Adjoint method for the slider-crank and HMMWV 13 models have been performed on the dual Intel Xeon computer and the quad AMD Opteron computer. Both parallel workstations have a shared memory architecture. POSIX threads (*p-threads*) [15] have been used as independent threads of computation. The parallel experiments have been performed using only a limited number of parallel processors (two and four, respectively). Therefore, the pipelined parallel Piecewise Adjoint method of Figs. 5.7 and 5.8 could not be tested effectively.

The execution times of both the slider-crank and the HMMWV 13 models show significant differences between the dual Intel Xeon computer and the quad AMD Opteron computer. This is explained not only by the different number of processors; i.e., two on to the dual Intel Xeon computer and four on the quad AMD Opteron computer, but also by the different types of processors and the different types of operating systems [54]; i.e., Windows XP on the dual Intel Xeon computer and Suse Linux 9.1 on the quad AMD Opteron computer.

The experiments with the implemented parallel Piecewise Adjoint method, on the limited parallel architectures that were available, show very low speed-ups or even slow-downs, when the number of independent threads is much larger than the number of parallel processors. However, given an adequate number of processors, the pipelined Piecewise Adjoint algorithm shows an improved predicted speed-up. Therefore, the

Piecewise Adjoint method may show benefits when the number of parallel processors is large enough to run the pipelined Piecewise Adjoint algorithm.

6.5.1 Parallel Experiments

Tables 6.18 and 6.19 and Figs. 6.5 and 6.6 present parallel and sequential results for the sensitivity of the slider-crank with 1, 2, 4, 8, and 16 parameters. Execution times are shown for the parallel Piecewise Adjoint method vs. sequential Direct Differentiation, Adjoint, and Piecewise Adjoint methods, on the dual Intel Xeon computer and the quad AMD Opteron computers, respectively. The simulated time interval for the slider-crank is $[0, 2]s$.

	1	2	4	8	16
DD	101.00	324.00	661.00	1436.00	2650.00
A	295.00	302.00	307.00	308.00	349.00
PA_{seq}	409.00	424.00	454.00	518.00	707.00
$PA_{parallel}$	297.00	313.00	349.00	393.00	536.00

Table 6.18: Parallel vs. sequential execution times for the slider-crank on the dual Intel Xeon computer

	1	2	4	8	16
DD	19.74	48.33	97.12	212.63	436.48
A	61.99	62.14	62.44	63.95	66.45
PA_{seq}	391.57	407.06	436.74	497.09	619.34
$PA_{parallel}$	152.14	169.5	177.98	195.06	243.49

Table 6.19: Parallel vs. sequential execution times for the slider-crank on the quad AMD Opteron computer

Tables 6.20 and 6.21 and Figs. 6.7 and 6.8 present parallel and sequential results for the sensitivity of the HMMWV 13 with 1, 2, 4, 8, and 16 parameters. Execution times are shown for the parallel Piecewise Adjoint method vs. sequential Direct Differentiation, Adjoint, and Piecewise Adjoint methods, on the dual Intel Xeon computer and the quad AMD Opteron computers, respectively. The simulated time interval for the HMMWV 13 is $[0, 0.05]s$.

	1	2	4	8	16
DD	146.00	235.00	397.00	811.00	1605.00
A	2149.00	2246.00	2279.00	2378.00	2435.00
PA_{seq}	57274.00	59279.00	63310.00	71540.00	88709.00
$PA_{parallel}$	150956.00	156994.00	174264.00	195175.00	261535.00

Table 6.20: Parallel vs. sequential execution times for the HMMWV 13 on the dual Intel Xeon computer

	1	2	4	8	16
DD	45.06	69.00	111.73	210.127	402.42
A	254.02	254.42	257.9	258.33	259.3
PA_{seq}	32436.92	34869.69	37755.71	43041.51	52510.64
$PA_{parallel}$	42168.72	46385.59	48704.87	55036.5	69346.00

Table 6.21: Parallel vs. sequential execution times for the HMMWV 13 on the quad AMD Opteron computer

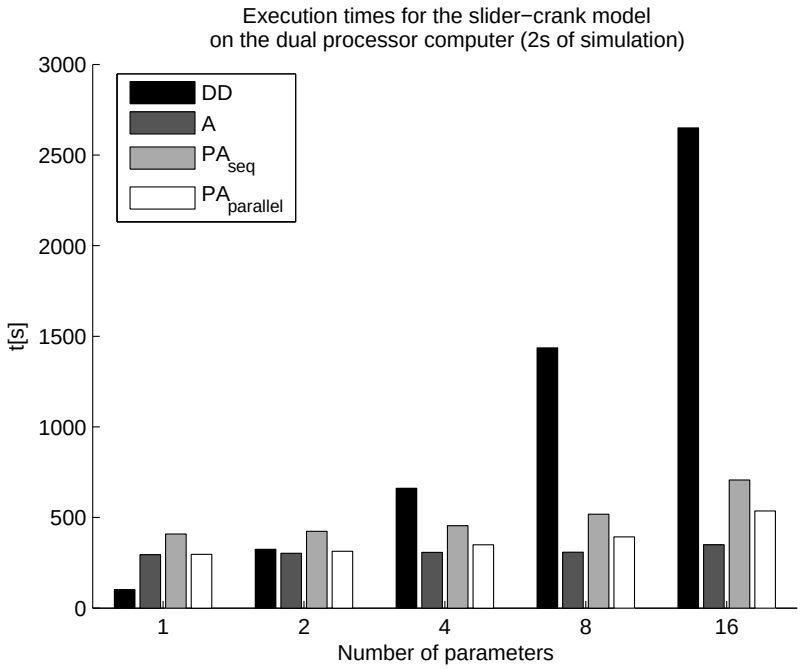

Figure 6.5: Execution times for the slider-crank model on the dual Intel Xeon computer

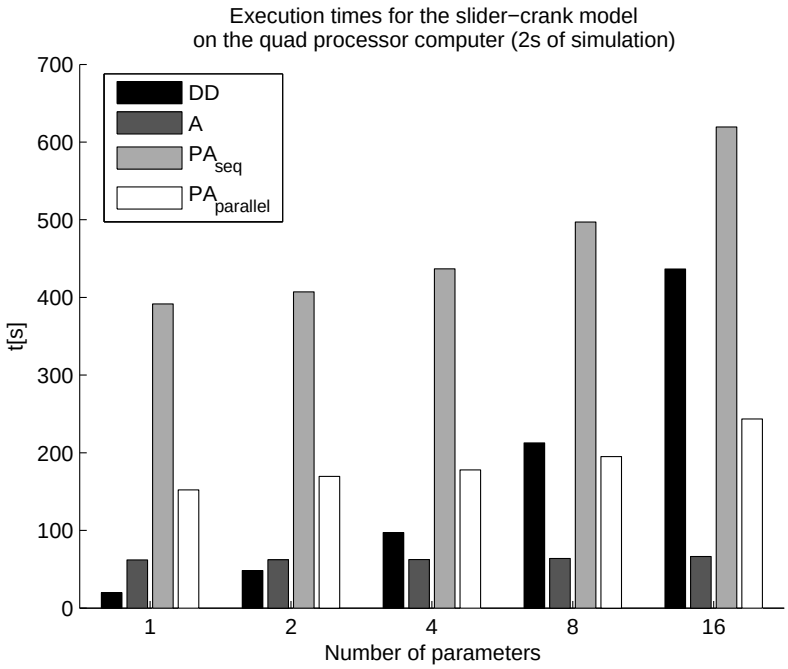

Figure 6.6: Execution times for the slider-crank model on the quad AMD Opteron computer

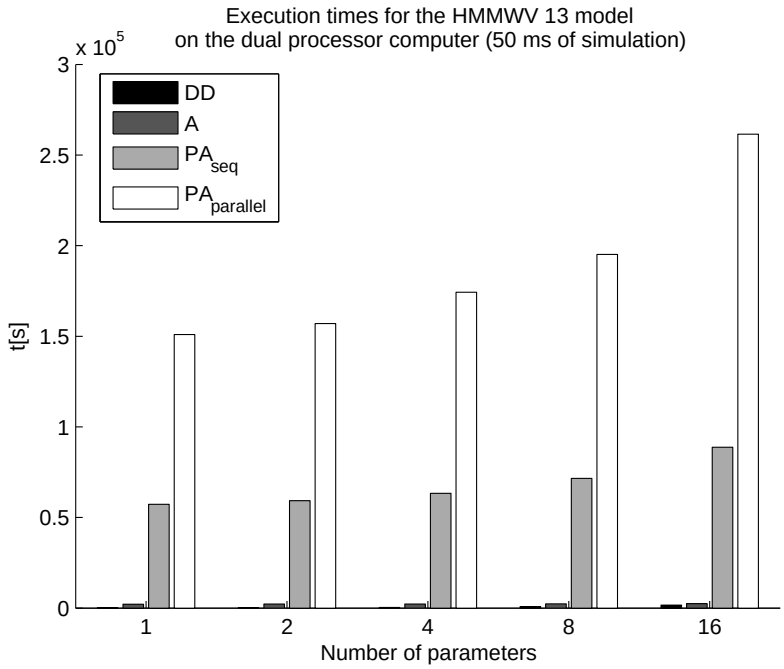

Figure 6.7: Execution time for the HMMWV 13 model on the dual Intel Xeon computer

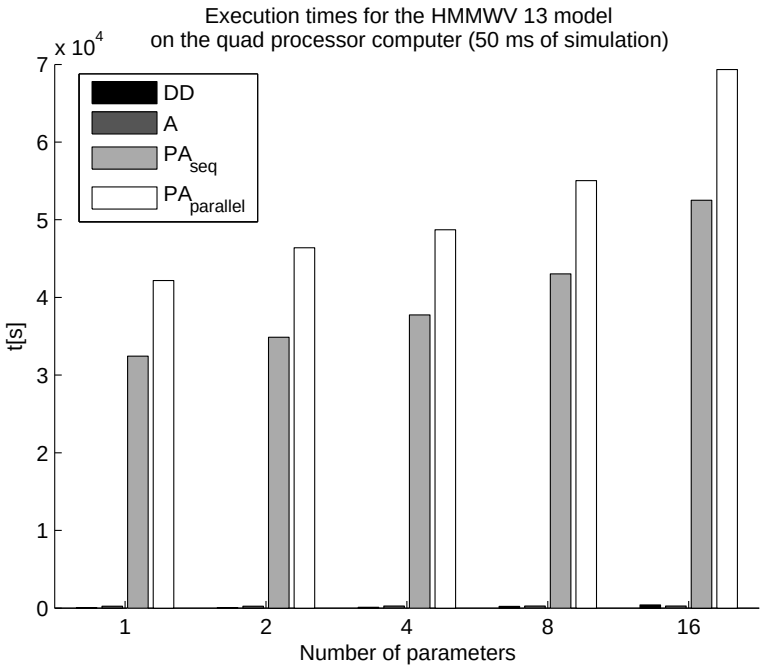

Figure 6.8: Execution time for the HMMWV 13 model on the quad AMD Opteron computer

It should be noted that the parallel experiments on the dual Intel Xeon and the quad AMD Opteron architectures could not take advantage of an exclusive use of the computers; i.e., the experiments were performed in a multi-tasking and multi-user environment [54] with other users running parallel tasks, too. Therefore, the execution times of Tables 6.18 through 6.21 may not accurately reflect the duration of the implemented parallel Piecewise Adjoint method on the corresponding parallel architectures. Also, there was not enough information available about the task load on each of the two parallel architectures to make a significant comparison between the execution times on the dual Intel Xeon and the quad AMD Opteron computers.

6.5.2 Analysis of Results of Experiments with the Sequential Piecewise Adjoint Method

On the dual Intel Xeon computer the data resulting from running the sequential Piecewise Adjoint method applied to the slider-crank, with $n = 28$, $m = 27$, $n_\beta \in \{1, 2, 4, 8, 16\}$, and $c = \frac{2}{3}$, on the $[0, 2]s$ interval, is the following:

1. for the integration of the DAE of motion $N_{steps}^{EOM} = 1110$, $N_J^{EOM} = 556$, $N_r^{EOM} = 1791$, $N_{Cf}^{EOM} = 0$ have been reported by the IDA integrator;

2. for the integration of one Adjoint DAE an average of $N_{steps}^{ADAE} = 294$, $N_J^{ADAE} = 25$, $N_r^{ADAE} = 534$, $N_{Cf}^{ADAE} = 0$ have been reported by the IDA integrator;

3. a total of $N_{grid} = 400$ mesh points have been used;

4. an average of $|\Delta^j|_{j=1,2,\ldots,N_j} = 10$ mesh points in the set of meshes $\{\Delta^j, j = 1, 2, \ldots, N_j\}$;

It should be noted that in addition to the DAE of motion, a number of $2d + 1$ Adjoint DAE are integrated. The DAE of motion and the Adjoint DAE do not depend on the design parameters n_β. Consequently, the integrator resulting from running the sequential Piecewise Adjoint method applied to a given model on a given architecture does not change with the number of design parameters n_β. However, the estimated number of flops of the sequential Piecewise Adjoint method does depend on the number of design parameters n_β, as it is shown by Eq. (5.151). The *predicted*

number of flops $N_{PA_{slider}}^{[0,2]}$ of the sequential Piecewise Adjoint method on the slider-crank is obtained by substituting for this data into Eq. (5.151).

On the dual Intel Xeon computer the data resulting from running the sequential Piecewise Adjoint method applied to the HMMWV 13, with $n = 91$, $m = 81$, $n_\beta \in \{1, 2, 4, 8, 16\}$, and $c = \frac{2}{3}$, on the $[0, 0.05]s$ interval, is the following:

1. for the integration of the DAE of motion $N_{steps}^{EOM} = 18$, $N_J^{EOM} = 5$, $N_r^{EOM} = 25$, $N_{Cf}^{EOM} = 0$ have been reported by the IDA integrator;

2. for the integration of one Adjoint DAE an average of $N_{steps}^{ADAE} = 93$, $N_J^{ADAE} = 42$, $N_r^{ADAE} = 128$, $N_{Cf}^{ADAE} = 12$ have been reported by the IDA integrator;

3. a total of $N_{grid} = 400$ mesh points have been used;

4. an average of $|\Delta^j|_{j=1,2,\ldots,N_j} = 10$ mesh points in the set of meshes $\{\Delta^j, j = 1, 2, \ldots, N_j\}$;

The predicted number of flops of the sequential Piecewise Adjoint method on the HMMWV 13 is obtained by substituting for this data into Eq. (5.151). Since the $[0, 2]s$ interval is 40 times larger than the $[0, 0.05]s$ interval, the estimated number of flops for the $[0, 2]s$ interval is $N_{PA_{HMMWV}}^{[0,2]} = 40 \times N_{PA_{HMMWV}}^{[0,0.05]}$.

Therefore, the *predicted ratio* of the execution time of sequential Piecewise Adjoint method applied to the HMMWV 13 model and the execution time of sequential Piecewise Adjoint method applied to the slider-crank model, on the dual Intel Xeon computer, is

$$\rho_{Xeon} = \frac{N_{PA_{HMMWV}}^{[0,2]}}{N_{PA_{slider}}^{[0,2]}}$$

For a number of design parameters $n_\beta \in \{1, 2, 4, 8, 16\}$, the average value of the predicted ratio is $\rho_{Xeon} = 4940$. Hence, the execution time of the sequential Piecewise Adjoint method applied to the HMMWV 13 model is therefore expected to be at least 4940 times more expensive than the sequential Piecewise Adjoint method applied to the slider-crank model, on the same interval of $[0, 2]s$. The results of the experiments with the two models on the dual Intel Xeon computer, presented in Tables 6.18 and 6.20 show that the *actual ratio* of the execution time of the sequential Piecewise Adjoint method applied to the HMMWV 13 and the execution time of the sequential Piecewise Adjoint method applied to the slider-crank is on average 5462.

On the quad AMD Opteron computer the data resulting from running the sequential Piecewise Adjoint method applied to the slider-crank, with $n = 28$, $m = 27$, $n_\beta \in \{1, 2, 4, 8, 16\}$, and $c = \frac{2}{3}$, on the $[0, 2]s$ interval, is the following:

1. for the integration of the DAE of motion $N_{steps}^{EOM} = 1108$, $N_J^{EOM} = 525$, $N_r^{EOM} = 1601$, $N_{Cf}^{EOM} = 0$ have been reported by the IDA integrator;

2. for the integration of one Adjoint DAE an average of $N_{steps}^{ADAE} = 270$, $N_J^{ADAE} = 22$, $N_r^{ADAE} = 525$, $N_{Cf}^{ADAE} = 0$ have been reported by the IDA integrator;

3. a total of $N_{grid} = 400$ mesh points have been used;

4. an average of $|\Delta^j|_{j=1,2,\ldots,N_j} = 10$ mesh points in the set of meshes $\{\Delta^j, j = 1, 2, \ldots, N_j\}$;

The predicted number of flops $N_{PA_{slider}}^{[0,2]}$ of the sequential Piecewise Adjoint method on the slider-crank is obtained by substituting for this data into Eq. (5.151).

On the quad AMD Opteron computer the data resulting from running the sequential Piecewise Adjoint method applied to the HMMWV 13, with $n = 91$, $m = 81$, $n_\beta \in \{1, 2, 4, 8, 16\}$, and $c = \frac{2}{3}$, on the $[0, 0.05]s$ interval, is the following:

1. for the integration of the DAE of motion $N_{steps}^{EOM} = 13$, $N_J^{EOM} = 4$, $N_r^{EOM} = 19$, $N_{Cf}^{EOM} = 0$ have been reported by the IDA integrator;

2. for the integration of one Adjoint DAE an average of $N_{steps}^{ADAE} = 59$, $N_J^{ADAE} = 32$, $N_r^{ADAE} = 95$, $N_{Cf}^{ADAE} = 7$ have been reported by the IDA integrator;

3. a total of $N_{grid} = 400$ mesh points have been used;

4. an average of $|\Delta^j|_{j=1,2,\ldots,N_j} = 10$ mesh points in the set of meshes $\{\Delta^j, j = 1, 2, \ldots, N_j\}$;

The predicted number of flops of the sequential Piecewise Adjoint method on the HMMWV 13 is obtained by substituting for this data into Eq. (5.151). Since the $[0, 2]s$ interval is 40 times larger than the $[0, 0.05]s$ interval, the estimated number of flops for the $[0, 2]s$ interval is $N_{PA}{}_{HMMWV}^{[0,2]} = 40 \times N_{PA}{}_{HMMWV}^{[0,0.05]}$.

Therefore, the predicted ratio of the execution time of sequential Piecewise Adjoint method applied to the HMMWV 13 model and the execution time of sequential Piecewise Adjoint method applied to the slider-crank model, on the quad AMD Opteron computer, is

$$\rho_{Opteron} = \frac{N_{PA}{}_{HMMWV}^{[0,2]}}{N_{PA}{}_{slider}^{[0,2]}}$$

For a number of design parameters $n_\beta \in \{1, 2, 4, 8, 16\}$, the average value of the predicted ratio is $\rho_{Opteron} = 3264$. Hence, the execution time of the sequential Piecewise

Adjoint method applied to the HMMWV 13 model is therefore expected to be at least 3264 times more expensive than the sequential Piecewise Adjoint method applied to the slider-crank model, on the same interval of $[0, 2]s$. The results of the experiments with the two models on the quad AMD Opteron computer, presented in Tables 6.19 and 6.21 show that the actual ratio of the execution time of the sequential Piecewise Adjoint method applied to the HMMWV 13 and the execution time of the sequential Piecewise Adjoint method applied to the slider-crank is on average 3410.

It should be noted that the predicted number of flops of the sequential Piecewise Adjoint method does not account for floating-point additions, integer operations, or the input-output operations in the intialization part of the algorithm. Hence, the prediction of the ratio of execution times for the sequential Piecewise Adjoint method for the two models is confirmed by the actual results on both the dual Intel Xeon and the quad AMD Opteron computers.

6.5.3 Analysis of Results of Experiments with the Parallel Piecewise Adjoint Method and Predictions of Speed-Up on an Adequate Multiprocessor Architecture

The parallel experiments with the slider-crank model on the dual Intel Xeon computer required 4 threads and only 2 processors were available. The parallel experiments with the slider-crank model on the quad AMD Opteron computer required

4 threads and 4 processors were available. For the slider-crank model the speed-up of the implemented parallel Piecewise Adjoint algorithm, defined as the ratio of the execution time of the sequential Piecewise Adjoint algorithm and the execution time of the implemented parallel Piecewise Adjoint algorithm, is on average 1.3 for the dual Intel Xeon computer and 2.5 for the quad AMD Opteron computer, as shown in Tables 6.18 and 6.19. According to Amdahl's law [29], the theoretical speed-up that can be obtained for computing task X on a computer with n_p processors is

$$S(n_p) = \frac{1}{1 - r + \frac{r}{n_p}} \tag{6.6}$$

where r is the percentage of the computing task X that is perfectly parallelizable; i.e., it does not require synchronization. Therefore, the maximum theoretical speed-up that can be achieved is $S_{max}(n_p) = n_p$, when $r = 100\%$. In practice, however, the parallelizable part of a computing task is never $r = 100\%$ and it always requires synchronization [15, 29].

For the slider-crank model with 16 parameters, an estimation of the speed-up of the pipelined Piecewise Adjoint algorithm relative to the sequential Piecewise Adjoint algorithm is obtained by substituting the data on the dual Intel Xeon computer into Eq. (5.164) and ignoring the synchronization time. Therefore, if a number of $n_{th}^{pipelined} = 15$ processors would be available, where $n_{th}^{pipelined}$ is defined by Eq. (5.165), then the predicted speed-up is

$$S_{slider}^{pipelined} = \frac{\tau_{PA}^{seq}}{\tau_{PA}^{pipelined}} \tag{6.7}$$

where τ_{PA}^{seq} is defined by Eq. (5.152) and $\tau_{PA}^{pipelined}$ is defined by Eq. (5.164). For

the data reported by the integrator when running the sequential Piecewise Adjoint algorithm, the predicted speed-up of the pipelined Piecewise Adjoint algorithm is

$$S_{slider}^{pipelined} = 5 \tag{6.8}$$

The parallel experiments with the HMMWV 13 model on the dual Intel Xeon computer required 22 threads and only 2 processors were available. The parallel experiments with the HMMWV 13 model on the quad AMD Opteron computer required 22 threads and only 4 processors were available. For the HMMWV 13 model the implemented parallel Piecewise Adjoint algorithm shows an average slow-down, defined as the ratio of the execution time of the implemented parallel Piecewise Adjoint algorithm and the sequential Piecewise Adjoint algorithm, of 2.8 on the dual Intel Xeon computer and 1.3 on the quad AMD Opteron computer, as shown in Tables 6.20 and 6.21. The slow-down seen in the HMMWV 13 model is due to the fact that the operating system is synchronizing $2d + 2 = 22$ threads on two processors on the dual Intel Xeon computer and on four processors on the quad AMD Opteron computer. When the number of threads is much larger than the number of available processors, the operating system spends a large amount of time scheduling the threads [54]. For the slider-crank model, which requires only $2d + 2 = 4$ threads, synchronization is substantially less expensive, especially on the quad AMD Opteron computer, which explains the speed-up seen when running the slider-crank parallel experiments.

For the HMMWV 13 model with 16 parameters, an estimation of the speed-up of the pipelined Piecewise Adjoint algorithm relative to the sequential Piecewise Adjoint algorithm is obtained by substituting the data on the dual Intel Xeon computer

into Eq. (5.164) and ignoring the synchronization time. Therefore, if a number of $n_{th}^{pipelined} = 33$ processors would be available, where $n_{th}^{pipelined}$ is defined by Eq. (5.165), then the predicted speed-up is

$$S_{HMMWV}^{pipelined} = \frac{\tau_{PA}^{seq}}{\tau_{PA}^{pipelined}} \tag{6.9}$$

where τ_{PA}^{seq} is defined by Eq. (5.152) and $\tau_{PA}^{pipelined}$ is defined by Eq. (5.164). For the data reported by the integrator when running the sequential Piecewise Adjoint algorithm, the predicted speed-up of the pipelined Piecewise Adjoint algorithm is

$$S_{HMMWV}^{pipelined} = 21 \tag{6.10}$$

CHAPTER 7
CONCLUSIONS AND RECOMMENDATIONS

7.1 Conclusions

The stability result of Chapter 3 shows that the backward integration of the index-3 Adjoint DAE is well-posed. This is an important theoretical result. In the past, there were only stability results for backward integration of Adjoint DAE of index up to two in Hessenberg form [16]. By showing stability of the index-3 Adjoint DAE of a multibody system, it follows that any underlying ODE or any lower index DAE obtained from the original index-3 Adjoint DAE through an index reduction method [13] has the same backward stability properties as the original. In particular, the Hiller-Anantharaman stabilized index-1 formulation [2] of the index-3 Adjoint DAE is backward stable.

Chapter 4 presents Hiller-Anantharaman stabilized index-1 formulations for the DAE of motion, Direct Differentiation DAE, and Adjoint DAE. Convergence of BDF methods applied to these formulations is proven, by showing that the resulting DAE are uniform index-1. It should be noted that Hiller and Anantharaman [2] did not prove convergence of any numerical integration method for the stabilized index-1 method they originally proposed.

Chapter 5 presents a new method, the Piecewise Adjoint method, that combines features of the Direct Differentiation method and the Adjoint method. The parallel computational structure and the forward time evolution of the Piecewise Ad-

joint method are also features seen in the Direct Differentiation method. The small number of DAE that need to be integrated in the Piecewise Adjoint method is also a feature seen in the Adjoint method. The Piecewise Adjoint method essentially is a multiple shooting [4] algorithm. In the Piecewise Adjoint method, the Adjoint problem is treated as a BVP, instead of a set of two IVP; i.e., the forward DAE of motion and the backward Adjoint DAE, as the classical multibody systems Adjoint method [26] is formulated.

The expression of the gradient of a functional of Eq. (5.103) is expanded as a function of the fundamental matrix and the particular solution of the linear Adjoint CPUODE and a set of unknown constant vectors. The fundamental matrix and the particular solution are obtained by backward integration of the Adjoint CPUODE, using the integrate-recover-assemble algorithm presented in Section 5.2 on a sequence of fine-grid time intervals that advance forward in time. After the final time is reached, the set of unknown vectors is solved for and the gradient is assembled. As a result, the Piecewise Adjoint method has a forward time evolution structure. Since the columns of the fundamental matrix and the particular solution can be integrated independently, the Piecewise Adjoint method has a coarse grained parallel computational structure.

An advantage of the Piecewise Adjoint method is that the fundamental matrix $Y_0^j(t)$, $t \in \mathcal{I}^j$, does not depend on the functional Ψ, since the columns of the fundamental matrix are obtained by integrating the homogeneous Adjoint DAE, through the integrate-recover-assemble algorithm. As a result, $Y_0^j(t)$ can be used for construct-

ing gradients of different functionals. Furthermore, the fundamental matrix can be used for solving the second order design sensitivity problem [26] through the Piecewise Adjoint method, because the homogeneous DAE of the second order Adjoint design sensitivity problem is the same as the homogeneous Adjoint DAE [26].

An efficiency analysis is performed by estimating the number of flops of the Piecewise Adjoint method. The efficiency analysis is used to predict the execution times of the sequential and Piecewise Adjoint algorithm and to predict the speed-up of the pipelined parallel Piecewise Adjoint method on multiprocessor architectures with adequate number of parallel processors. Also, a memory load analysis shows an advantage of the Piecewise Adjoint method over the Adjoint method.

The Piecewise Adjoint method may become more efficient than the Direct Differentiation method, if the number of design parameters is much larger than $2d+2$ and the number of available processors is close to $2d + 2$, where d represents the degrees of freedom [27] of the multibody system. The advantage of the Piecewise Adjoint method over the Adjoint method is a decreased memory load, resulting from the overall forward time evolution of the algorithm, as shown in Section 5.6. The disadvantage of the Piecewise Adjoint method relative to the Adjoint method is the increased execution time. If the number of processors is too small relative to the number $2d + 2$ of necessary threads then the parallel Piecewise Adjoint method may become too expensive. The parallel and sequential numerical experiments confirm the efficiency analysis predictions.

On a computer architecture with a sufficient number of parallel processors,

the pipelined Piecewise Adjoint algorithm of Figs. 5.7 and 5.8 may be used. It requires an additional number of $|\Delta^j|_{j=1,2,\ldots,N_j}+1$ processors, where $|\Delta^j|_{j=1,2,\ldots,N_j}$ is the average number of mesh points in the set of meshes $\{\Delta^j, j=1,2,\ldots,N_j\}$. The efficiency analysis, which is validated by the numerical experiments with the sequential Piecewise Adjoint algorithm and the implemented parallel Piecewise Adjoint algorithm, shows an increased theoretical speed-up for the pipelined Piecewise Adjoint algorithm. The speed-up of the the pipelined Piecewise Adjoint algorithm relative to the sequential Piecewise Adjoint algorithm, whose estimation for the HMMWV 13 model is shown by Eq. (6.10), can be substantially larger than the speed-up of the implemented Piecewise Adjoint algorithm. Although this estimation does not account for the synchronization time, if there are enough processors; e.g., 33 for the HMMWV 13 model, the amount spent on synchronization may be minimized [1, 19].

7.2 Recommended Future Work

The fundamental matrices computed through the Piecewise Adjoint method can be used to solve the linear DAE resulting from the second order design sensitivity analysis of multibody dynamics [26]. Therefore, extending the Piecewise Adjoint method to the second order design sensitivity analysis may increase the fraction of the computing effort that can benefit from parallelism. As a result, the Piecewise Adjoint method may provide some efficiency gain for the second order design sensitivity analysis.

On the practical side, it is worth implementing the pipelined version of the

Piecewise Adjoint method doing more careful experiments with the algorithm and including parallel architectures with more processors and perhaps experimenting how the approach would perform on a distributed-memory parallel architecture [45]. A comparison between the shared memory and the distributed memory parallel architectures may provide information on how well the two parallel architectures scale with the size of the problem.

Taking advantage of the cache memory architecture can be a powerful method for increasing efficiency of the numerical algorithms [29]. Therefore, memory cache management and adapting the numerical algorithms to a given cache architecture should be investigated.

APPENDIX A

A.1 Evaluation of Derivatives of Matrices

Consider a $n \times m$ matrix function $X(w(t), t)$ depending explicitly on independent variable t and implicitly through vector $w(t)$,

$$X(w(t), t) = \begin{pmatrix} x_1(w(t), t) & \dots & x_m(w(t), t) \end{pmatrix} \tag{A.1}$$

where $x_i(w(t), t)$ is the i-th column of matrix X, $i = 1, 2, \dots, m$. Assume that X has continuous first and second partial derivatives. Differentiation of X with respect to independent variable t yields

$$\frac{dX(w(t), t)}{dt} = \begin{pmatrix} \frac{dx_1(w(t),t)}{dt} & \dots & \frac{dx_m(w(t),t)}{dt} \end{pmatrix} \tag{A.2}$$

Applying the chain rule of differentiation, the derivative of each column x_i, $i = 1, 2, \dots, m$, is

$$x_i{}' = \frac{dx_i(w(t), t)}{dt} = \frac{\partial x_i}{\partial w}\frac{dw(t)}{dt} + \frac{\partial x_i}{\partial t} = x_{iw}w' + x_{it} \tag{A.3}$$

where x_{iw} represents the partial differentiation of vector x_i with respect to vector w and x_{it} represents the partial differentiation of vector x_i with respect to independent variable t. Substituting $x_i{}'$, $i = 1, 2, \dots, m$, in Eq. (A.2),

$$X' = \frac{dX(w(t), t)}{dt} = \begin{pmatrix} x_{1w}w' + x_{1t} & \dots & x_{mw}w' + x_{mt} \end{pmatrix} \tag{A.4}$$

Let u_i be the m-dimensional unit vector with all elements zero except the i-th element, which is one. Multiplying matrix X by unit vector u_i

$$Xu_i = x_i \tag{A.5}$$

As a result,

$$\begin{aligned} {x_i}' &= x_{iw}w' + x_{it} \\ &= (Xu_i)_w w' + (Xu_i)_t \end{aligned} \tag{A.6}$$

Substituting the derivative of matrix x_i of Eq. (A.6) into Eq. (A.4), the first derivative of X is re-written as

$$X' = \begin{pmatrix} (Xu_1)_w w' + (Xu_1)_t & \dots & (Xu_m)_w w' + (Xu_m)_t \end{pmatrix} \tag{A.7}$$

In order to evaluate the second derivative of X, Eq. (A.7) is differentiated,

$$\frac{d^2X(w(t),t)}{dt^2} = \begin{pmatrix} \frac{dx_1{}'}{dt} & \dots & \frac{dx_m{}'}{dt} \end{pmatrix} \tag{A.8}$$

where each column ${x_i}'$, $i = 1, 2, \dots, m$, is a function of w, w', and t of the form of Eq. (A.3). Applying the chain rule of differentiation,

$${x_i}'' = \frac{d}{dt}({x_i}'(w(t), w'(t), t)) = ({x_i}')_w w' + ({x_i}')_{w'} w'' + ({x_i}')_t \tag{A.9}$$

Substituting ${x_i}'$ of Eq. (A.3) into Eq. (A.9),

$$\begin{aligned} {x_i}'' &= ((x_i)_w w' + (x_i)_t)_w w' \\ &+ ((x_i)_w w' + (x_i)_t)_{w'} w'' \\ &+ ((x_i)_w (\hat{w'}) + (x_i)_t)_t \end{aligned} \tag{A.10}$$

where terms of the form $(A(\eta)\hat{\xi})_\eta$ indicate that variable ξ is treated as a constant for differentiation with respect to variable η. For a function x_i with continuous second partial derivatives, $(x_i)_{t,w} = (x_i)_{w,t}$. Moreover, $((x_i)_t)_{w'} = 0$, since x_i does not depend

on w', and $\left(\left(x_i\right)_w w'\right)_{w'} = \left(x_i\right)_w$. As a result, ${x_i}''$ is re-written as

$$ {x_i}'' = \left(\left(x_i\right)_w w'\right)_w w' + \left(x_i\right)_w w'' + 2\left(x_i\right)_{w,t} w' + \left(x_i\right)_{t,t} \tag{A.11} $$

Therefore,

$$ X'' = \begin{pmatrix} {x_1}'' & \dots & {x_m}'' \end{pmatrix} \tag{A.12} $$

where, using Eq. (A.5),

$$ {x_i}'' = \left(\left(Xu_i\right)_w w'\right)_w w' + \left(Xu_i\right)_w w'' + 2\left(Xu_i\right)_{w,t} w' + \left(Xu_i\right)_{t,t} \tag{A.13} $$

A.2 Differentiation of the Constraint Jacobian

Φ_q

Let the constraint function be

$$ \Phi(q, \beta, t) = \begin{pmatrix} \varphi_1 \\ \vdots \\ \varphi_m \end{pmatrix} \tag{A.14} $$

where

$$ q(\beta, t) = \begin{pmatrix} {q_1}^\top & \dots & {q_n}^\top \end{pmatrix}^\top $$

is the generalized coordinate depending on the vector of design parameters β and time t. Therefore, the constraint Jacobian is

$$ \Phi_q = \begin{pmatrix} \varphi_{1_{q_1}} & \dots & \varphi_{1_{q_n}} \\ \vdots & & \vdots \\ \varphi_{m_{q_1}} & \dots & \varphi_{m_{q_n}} \end{pmatrix} = \begin{pmatrix} \Phi_{q_1} & \dots & \Phi_{q_n} \end{pmatrix} \tag{A.15} $$

where

$$\Phi_{q_i} = \begin{pmatrix} \varphi_{1_{q_i}} \\ \vdots \\ \varphi_{m_{q_i}} \end{pmatrix}, i = 1, 2, \ldots, n \tag{A.16}$$

In order to calculate the first derivative of the constraint Jacobian, the identity of Eq. (A.4) is used, where $X \equiv \Phi_q$, $w \equiv q$, and $x_i \equiv \Phi_{q_i}$, $i = 1, 2, \ldots, n$. As a result,

$$\begin{aligned} \Phi_q{}' &= \begin{pmatrix} \Phi_{q_1 q} q' + \Phi_{q_1 t} & \ldots & \Phi_{q_n q} q' + \Phi_{q_n t} \end{pmatrix} \\ &= \begin{pmatrix} \Phi_{q_1 q} q' & \ldots & \Phi_{q_n q} q' \end{pmatrix} + \Phi_{q,t} \end{aligned} \tag{A.17}$$

where

$$\begin{aligned} \Phi_{q,t} &= \begin{pmatrix} \Phi_{q_1 t} & \ldots & \Phi_{q_n t} \end{pmatrix} \\ &= \begin{pmatrix} \varphi_{1_{q_1,t}} & \cdots & \varphi_{1_{q_n,t}} \\ \vdots & & \vdots \\ \varphi_{m_{q_1,t}} & \cdots & \varphi_{m_{q_n,t}} \end{pmatrix} \end{aligned} \tag{A.18}$$

is the partial derivative of the constraint Jacobian with respect to independent variable t. Using the notation

$$\begin{aligned} \Phi_{q,q}(\gamma) &\equiv \begin{pmatrix} \Phi_{q_1 q}\gamma & \ldots & \Phi_{q_n q}\gamma \end{pmatrix} \\ &= \begin{pmatrix} \varphi_{1_{q_1,q}}\gamma & \cdots & \varphi_{1_{q_n,q}}\gamma \\ \vdots & & \vdots \\ \varphi_{m_{q_1,q}}\gamma & \cdots & \varphi_{m_{q_n,q}}\gamma \end{pmatrix} \end{aligned} \tag{A.19}$$

where γ is a n-dimensional vector, the derivative of the constraint Jacobian of Eq. (A.17) is re-written as

$$\Phi_q{}' = \Phi_{q,q}(q') + \Phi_{q,t} \tag{A.20}$$

Proposition A.1. *Let γ be a n-dimensional vector that does not depend on q. If $\Phi(q,\beta,t)$ is at least twice continously differentiable with respect to q_i, $i=1,2,\ldots,n$, then matrix $\Phi_{q,q}(\gamma)$ has the property*

$$\Phi_{q,q}(\gamma) = \left(\Phi_q\gamma\right)_q$$

Proof.

$$\begin{aligned}
\left(\Phi_q\gamma\right)_q &= \frac{\partial}{\partial q}\left(\begin{pmatrix} \varphi_{1_{q_1}} & \cdots & \varphi_{1_{q_n}} \\ \vdots & & \vdots \\ \varphi_{m_{q_1}} & \cdots & \varphi_{m_{q_n}} \end{pmatrix}\gamma\right) = \frac{\partial}{\partial q}\left(\begin{pmatrix} \varphi_{1_q} \\ \vdots \\ \varphi_{m_q} \end{pmatrix}\gamma\right) \\
&= \frac{\partial}{\partial q}\begin{pmatrix} \varphi_{1_q}\gamma \\ \vdots \\ \varphi_{m_q}\gamma \end{pmatrix} = \begin{pmatrix} (\varphi_{1_q}\gamma)_q \\ \vdots \\ (\varphi_{m_q}\gamma)_q \end{pmatrix} \\
&= \begin{pmatrix} (\varphi_{1_q}\gamma)_{q_1} & \cdots & (\varphi_{1_q}\gamma)_{q_n} \\ \vdots & & \vdots \\ (\varphi_{m_q}\gamma)_{q_1} & \cdots & (\varphi_{m_q}\gamma)_{q_n} \end{pmatrix} = \begin{pmatrix} \varphi_{1_{q,q_1}}\gamma & \cdots & \varphi_{1_{q,q_n}}\gamma \\ \vdots & & \vdots \\ \varphi_{m_{q,q_1}}\gamma & \cdots & \varphi_{m_{q,q_n}}\gamma \end{pmatrix}
\end{aligned} \tag{A.21}$$

Since Φ is at least twice continously differentiable with respect to variables q_i, $i=1,2,\ldots,n$, the following identities hold [44]:

$$\frac{\partial^2\varphi_j}{\partial q_i \partial q_k} = \frac{\partial^2\varphi_j}{\partial q_k \partial q_i}$$

for $i, k \in \{1, 2, \ldots, n\}$ and $j \in \{1, 2, \ldots, m\}$. Therefore, $\varphi_{j_{q,q_i}} = \varphi_{j_{q_i,q}}$. Hence,

$$\begin{aligned}
\left(\Phi_q \gamma\right)_q &= \begin{pmatrix} \varphi_{1_{q,q_1}}\gamma & \cdots & \varphi_{1_{q,q_n}}\gamma \\ \vdots & & \vdots \\ \varphi_{m_{q,q_1}}\gamma & \cdots & \varphi_{m_{q,q_n}}\gamma \end{pmatrix} \\
&= \begin{pmatrix} \varphi_{1_{q_1,q}}\gamma & \cdots & \varphi_{1_{q_n,q}}\gamma \\ \vdots & & \vdots \\ \varphi_{m_{q_1,q}}\gamma & \cdots & \varphi_{m_{q_n,q}}\gamma \end{pmatrix} = \Phi_{q,q}(\gamma)
\end{aligned} \tag{A.22}$$

■

Substituting for $\Phi_{q,q}(q')$ by $\left(\Phi_q q'\right)_q$, the derivative of the constraint Jacobian is

$$\Phi_q{}' = \left(\Phi_q q'\right)_q + \Phi_{q,t} \tag{A.23}$$

The second derivative of the constraint Jacobian is obtained using Eq. (A.12), where $X \equiv \Phi_q$, $w \equiv q$, and $x_i \equiv \Phi_{q_i}$, $i = 1, 2, \ldots, n$. Thus,

$$\Phi_q{}'' = \begin{pmatrix} \Phi_{q_1}{}'' & \ldots & \Phi_{q_n}{}'' \end{pmatrix} \tag{A.24}$$

According to Eq. (A.11),

$$\Phi_{q_i}{}'' = \left(\left(\Phi_{q_i}\right)_q q'\right)_q q' + \left(\Phi_{q_i}\right)_q q'' + 2\left(\Phi_{q_i}\right)_{q,t} q' + \left(\Phi_{q_i}\right)_{t,t} \tag{A.25}$$

for $i = 1, 2, \ldots, n$. Substituting for $\Phi_{q_i}{}''$ in Eq. (A.24),

$$\begin{aligned}
\Phi_q{}'' &= \begin{pmatrix} \left(\left(\Phi_{q_1}\right)_q q'\right)_q q' & \ldots & \left(\left(\Phi_{q_n}\right)_q q'\right)_q q' \end{pmatrix} \\
&+ \begin{pmatrix} \left(\Phi_{q_1}\right)_q q'' & \ldots & \left(\Phi_{q_n}\right)_q q'' \end{pmatrix} \\
&+ 2\begin{pmatrix} \left(\Phi_{q_1}\right)_{q,t} q' & \ldots & \left(\Phi_{q_n}\right)_{q,t} q' \end{pmatrix} \\
&+ \begin{pmatrix} \left(\Phi_{q_1}\right)_{t,t} & \ldots & \left(\Phi_{q_n}\right)_{t,t} \end{pmatrix}
\end{aligned} \tag{A.26}$$

Defining matrices

$$\Phi_{q,q,q}(\gamma,\zeta) \equiv \begin{pmatrix} ((\Phi_{q_1})_q\gamma)_q\zeta & \dots & ((\Phi_{q_n})_q\gamma)_q\zeta \end{pmatrix} \quad (A.27)$$

$$\Phi_{q,q,q''} \equiv \begin{pmatrix} (\Phi_{q_1})_q q'' & \dots & (\Phi_{q_n})_q q'' \end{pmatrix} \quad (A.28)$$

$$\Phi_{q,q,t} \equiv \begin{pmatrix} (\Phi_{q_1})_{q,t} q' & \dots & (\Phi_{q_n})_{q,t} q' \end{pmatrix} \quad (A.29)$$

$$\Phi_{q,t,t} \equiv \begin{pmatrix} (\Phi_{q_1})_{t,t} & \dots & (\Phi_{q_n})_{t,t} \end{pmatrix} \quad (A.30)$$

$$(A.31)$$

where γ and ζ are n-dimensional vectors, the second derivative of the Jacobian is re-written as

$$\Phi_q{}'' = \Phi_{q,q,q}(q',q') + \Phi_{q,q,q''} + 2\Phi_{q,q,t} + \Phi_{q,t,t} \quad (A.32)$$

Matrix $\Phi_{q,t,t}$, of Eq. (A.30),

$$\Phi_{q,t,t} = \begin{pmatrix} (\Phi_{q_1})_{t,t} & \dots & (\Phi_{q_n})_{t,t} \end{pmatrix} = \frac{\partial^2}{\partial t^2}(\Phi_q) \quad (A.33)$$

is the second partial derivative of the Jacobian with respect to time. For a general matrix $A(t)$ and vector $\gamma(t)$, $(\frac{\partial}{\partial t}(A(t)))\gamma(t) = \frac{\partial}{\partial t}(A(t)\hat{\gamma})$. Therefore,

$$\begin{aligned} \Phi_{q,q,t} &= \begin{pmatrix} \frac{\partial}{\partial t}((\Phi_{q_1})_q(\hat{q'})) & \dots & \frac{\partial}{\partial t}((\Phi_{q_n})_q(\hat{q'})) \end{pmatrix} \\ &= \frac{\partial}{\partial t}\left(\begin{pmatrix} (\Phi_{q_1})_q(\hat{q'}) & \dots & (\Phi_{q_n})_q(\hat{q'}) \end{pmatrix}\right) \\ &= \frac{\partial}{\partial t}(\Phi_{q,q}((\hat{q'}))) \end{aligned} \quad (A.34)$$

Substituting for $\Phi_{q,q}((\hat{q'}))$ by its expression given in Proposition A.1, $\Phi_{q,q,t}$ is re-written as

$$\Phi_{q,q,t} = \frac{\partial}{\partial t}((\Phi_q(\hat{q'}))_q) = (\Phi_q(\hat{q'}))_{q,t} \quad (A.35)$$

Matrix $\Phi_{q,q,q''}$, defined by Eq. (A.28), is

$$\Phi_{q,q,q''} = \left(\begin{array}{ccc} (\Phi_{q_1})_q q'' & \dots & (\Phi_{q_n})_q q'' \end{array} \right) = \Phi_{q,q}(q'') \tag{A.36}$$

Using the identity proved in Proposition A.1,

$$\Phi_{q,q,q''} = \Phi_{q,q}(q'') = (\Phi_q q'')_q \tag{A.37}$$

Proposition A.2. *Let γ and ζ be n-dimensional vectors not depending on q. If $\Phi(q, \beta, t)$ is at least three times continously differentiable with respect to variables q_i, $i = 1, 2, \dots, n$, then matrix $\Phi_{q,q,q}(\gamma, \zeta)$ has the property*

$$\Phi_{q,q,q}(\gamma, \zeta) = ((\Phi_q \gamma)_q \zeta)_q$$

Proof.

$$\begin{aligned} ((\Phi_q\gamma)_q\zeta)_q &= \left(\begin{array}{ccc} ((\Phi_q\gamma)_q\zeta)_{q_1} & \dots & ((\Phi_q\gamma)_q\zeta)_{q_n} \end{array} \right) \\ &= \left(\begin{array}{ccc} (\Phi_q\gamma)_{q,q_1}\zeta & \dots & (\Phi_q\gamma)_{q,q_n}\zeta \end{array} \right) \\ &= \left(\begin{array}{ccc} (\Phi_{q,q_1}\gamma)_q\zeta & \dots & (\Phi_{q,q_n}\gamma)_q\zeta \end{array} \right) \\ &= \left(\begin{array}{ccc} (\Phi_{q_1,q}\gamma)_q\zeta & \dots & (\Phi_{q_n,q}\gamma)_q\zeta \end{array} \right) \\ &= \left(\begin{array}{ccc} ((\Phi_{q_1})_q\gamma)_q\zeta & \dots & ((\Phi_{q_n})_q\gamma)_q\zeta \end{array} \right) = \Phi_{q,q,q}(\gamma, \zeta) \end{aligned} \tag{A.38}$$

Interchanging the differentiation order $\Phi_{q,q_i} = \Phi_{q_i,q}$, $i = 1, 2, \dots, n$, is possible because function Φ is assumed to be more than twice continously differentiable. ■

Substituting for $\Phi_{q,q,q}(q', q')$ the expression $((\Phi_q q')_q q')_q$, according to Proposition A.2; for $\Phi_{q,q,q''}$ the expression of Eq. (A.37); for $\Phi_{q,q,t}$ the expression of Eq. (A.35); and for $\Phi_{q,t,t}$ the expression of Eq. (A.33) in Eq. (A.32), the second derivative

of the Jacobian is

$$\Phi_q{}'' = ((\Phi_q q')_q q')_q + (\Phi_q q'')_q + 2(\Phi_q \hat{q}')_{q,t} + \frac{\partial^2}{\partial t^2}(\Phi_q) \tag{A.39}$$

The Hessian $(\varphi_{i_q}{}^\top)_q$ of $\varphi_i, i = 1, 2, \ldots, m$, is symmetric, $(\varphi_{i_q}{}^\top)_q = ((\varphi_{i_q}{}^\top)_q)^\top$, and the Hessian $(\varphi_{i_{q,t}}{}^\top)_q$ of $\varphi_{it} \equiv \frac{\partial \varphi_i}{\partial t}, i = 1, 2, \ldots, m$, is also symmetric. As a result, the following identities hold:

$$\begin{aligned}
\Phi_{q,q}(q') = (\Phi_q q')_q &= \left(\begin{pmatrix} \varphi_{1_q} \\ \vdots \\ \varphi_{m_q} \end{pmatrix} q' \right)_q = \begin{pmatrix} \varphi_{1_q} q' \\ \vdots \\ \varphi_{m_q} q' \end{pmatrix}_q = \begin{pmatrix} (q'^\top \varphi_{1_q}{}^\top)_q \\ \vdots \\ (q'^\top \varphi_{m_q}{}^\top)_q \end{pmatrix} \\
&= \begin{pmatrix} q'^\top (\varphi_{1_q}{}^\top)_q \\ \vdots \\ q'^\top (\varphi_{m_q}{}^\top)_q \end{pmatrix} = \begin{pmatrix} q'^\top ((\varphi_{1_q}{}^\top)_q)^\top \\ \vdots \\ q'^\top ((\varphi_{m_q}{}^\top)_q)^\top \end{pmatrix} = \Theta
\end{aligned} \tag{A.40}$$

where Θ was introduced in Eq. (3.63) of Chapter 3;

$$(\Phi_{q,q}(q'))_t = (\Phi_q q')_{q,t} = ((\Phi_q q')_q)_t = \Theta_t \tag{A.41}$$

$$\begin{aligned}
\Phi_{q,q}(q'') = (\Phi_q q'')_q &= \left(\begin{pmatrix} \varphi_{1_q} \\ \vdots \\ \varphi_{m_q} \end{pmatrix} q'' \right)_q = \begin{pmatrix} \varphi_{1_q} q'' \\ \vdots \\ \varphi_{m_q} q'' \end{pmatrix}_q = \begin{pmatrix} (q''^\top \varphi_{1_q}{}^\top)_q \\ \vdots \\ (q''^\top \varphi_{m_q}{}^\top)_q \end{pmatrix} \\
&= \begin{pmatrix} q''^\top (\varphi_{1_q}{}^\top)_q \\ \vdots \\ q''^\top (\varphi_{m_q}{}^\top)_q \end{pmatrix} = \begin{pmatrix} q''^\top ((\varphi_{1_q}{}^\top)_q)^\top \\ \vdots \\ q''^\top ((\varphi_{m_q}{}^\top)_q)^\top \end{pmatrix} = Z
\end{aligned} \tag{A.42}$$

where Z was introduced in Eq. (3.86) of Chapter 3; and

$$\begin{aligned}\Phi_{q,q,t}q' = (\Phi_{q,t}q')_q &= \left(\begin{pmatrix}\varphi_{1_{q,t}}\\ \vdots \\ \varphi_{m_{q,t}}\end{pmatrix}q'\right)_q = \begin{pmatrix}\varphi_{1_{q,t}}q'\\ \vdots \\ \varphi_{m_{q,t}}q'\end{pmatrix}_q = \begin{pmatrix}(q'^{\top}\varphi_{1_{q,t}}{}^{\top})_q\\ \vdots \\ (q'^{\top}\varphi_{m_{q,t}}{}^{\top})_q\end{pmatrix}\\ &= \begin{pmatrix}q'^{\top}(\varphi_{1_{q,t}}{}^{\top})_q\\ \vdots \\ q'^{\top}(\varphi_{m_{q,t}}{}^{\top})_q\end{pmatrix} = \begin{pmatrix}q'^{\top}((\varphi_{1_{q,t}}{}^{\top})_q)^{\top}\\ \vdots \\ q'^{\top}((\varphi_{m_{q,t}}{}^{\top})_q)^{\top}\end{pmatrix} = T \qquad \text{(A.43)}\end{aligned}$$

where T was introduced in Eq. (3.82) of Chapter 3. Let $\theta_i = q'^{\top}((\varphi_{i_q}{}^{\top})_q)^{\top}$ be a row-vector of matrix Θ, $i \in \{1, 2, \ldots, m\}$. Then,

$$\begin{aligned}(\theta_i{}^{\top})_q &= ((\varphi_{i_q}{}^{\top})_q q')_q = \left(\begin{pmatrix}\varphi_{i_{q_1,q_1}} & \cdots & \varphi_{i_{q_1,q_n}}\\ \vdots & \ddots & \vdots \\ \varphi_{i_{q_n,q_1}} & \cdots & \varphi_{i_{q_n,q_n}}\end{pmatrix}\begin{pmatrix}q_1{}'\\ \vdots \\ q_n{}'\end{pmatrix}\right)_q\\ &= \begin{pmatrix}\sum_{j=1}^{n}\varphi_{i_{q_1,q_j,q_1}}q_j{}' & \cdots & \sum_{j=1}^{n}\varphi_{i_{q_1,q_j,q_n}}q_j{}'\\ \vdots & \ddots & \vdots \\ \sum_{j=1}^{n}\varphi_{i_{q_n,q_j,q_1}}q_j{}' & \cdots & \sum_{j=1}^{n}\varphi_{i_{q_n,q_j,q_n}}q_j{}'\end{pmatrix} \qquad \text{(A.44)}\end{aligned}$$

and, since $\varphi_{i_{q_k,q_j,q_l}} = \varphi_{i_{q_l,q_j,q_k}}, k, l = 1, 2, \ldots, n$, matrix $(\theta_i{}^{\top})_q$ is symmetric; i.e.,

$(\theta_i{}^\top)_q = ((\theta_i{}^\top)_q)^\top$. Therefore,

$$
\begin{aligned}
((\Phi_q q')_q q')_q = (\Theta q')_q &= \left(\begin{pmatrix} \theta_1 \\ \vdots \\ \theta_m \end{pmatrix} q' \right)_q = \begin{pmatrix} \theta_1 q' \\ \vdots \\ \theta_m q' \end{pmatrix}_q = \begin{pmatrix} (q'^\top \theta_1{}^\top)_q \\ \vdots \\ (q'^\top \theta_m{}^\top)_q \end{pmatrix} \\
&= \begin{pmatrix} q'^\top (\theta_1{}^\top)_q \\ \vdots \\ q'^\top (\theta_m{}^\top)_q \end{pmatrix} = \begin{pmatrix} q'^\top ((\theta_1{}^\top)_q)^\top \\ \vdots \\ q'^\top ((\theta_m{}^\top)_q)^\top \end{pmatrix} = \Upsilon
\end{aligned} \tag{A.45}
$$

where Υ was introduced in Eq. (3.76) of Chapter 3.

As a result of the identities of Eqs. (A.40) through (A.45), the following identities hold:

$$
\begin{aligned}
K_{21} &\equiv \frac{d}{dt}(\Phi_q) = \Phi_{q,t} + (\Phi_q q')_q = \Phi_{q,t} + \Theta \\
&= \begin{pmatrix} \Phi_{u,t} & \Phi_{v,t} \end{pmatrix} + \begin{pmatrix} \Theta^u & \Theta^v \end{pmatrix} = \begin{pmatrix} \Phi_{u,t} + \Theta^u & \Phi_{v,t} + \Theta^v \end{pmatrix}
\end{aligned} \tag{A.46}
$$

$$
\begin{aligned}
K_{31} &\equiv \frac{d^2}{dt^2}(\Phi_q) = \Phi_{q,tt} + (\Phi_{q,t} q')_q + (\Phi_q \hat{q'})_{q,t} + ((\Phi_q q')_q q')_q + (\Phi_q q'')_q \\
&= \Phi_{q,tt} + T + \Theta_t + \Upsilon + Z \\
&= \begin{pmatrix} \Phi_{u,tt} + T^u + \Theta_t{}^u + \Upsilon^u + Z^u & \Phi_{v,tt} + T^v + \Theta_t{}^v + \Upsilon^v + Z^v \end{pmatrix}
\end{aligned} \tag{A.47}
$$

If the constraint function Φ is at least twice continously differentiable with respect to q and t, then $T = \Theta_t$; i.e.,

$$
(\Phi_{q,t} q')_q = (\Phi_q \hat{q'})_{q,t} \tag{A.48}
$$

Consider a partitioning $q = \begin{pmatrix} u_{m\times 1}{}^\top & v_{d\times 1}{}^\top \end{pmatrix}^\top$ of the generalized coordinate vector q into dependent part u and independent part v, such that the multibody

system constraint Jacobian Φ_q is correspondingly partitioned as $\begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix}$, with Φ_u non-singular. Since the constraint Jacobian is assumed to have full row-rank at all times, such a partition always exists [58]. In general, there is a $m \times m$ row permutation constant matrix P_r and a $n \times n$ column permutation constant matrix P_c, such that [58]

$$P_r \Phi_q P_c = \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} \tag{A.49}$$

Define $n \times d$ matrix

$$X_0 = \begin{pmatrix} -{\Phi_u}^{-1}\Phi_v \\ I_d \end{pmatrix} \tag{A.50}$$

and $n \times m$ matrix

$$X_1 = \begin{pmatrix} {\Phi_u}^{-1} \\ 0 \end{pmatrix} \tag{A.51}$$

In order to compute derivatives of matrix X_0, the identity

$$P_r \Phi_q P_c X_0 = \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} \begin{pmatrix} -{\Phi_u}^{-1}\Phi_v \\ I_d \end{pmatrix} = 0 \tag{A.52}$$

is differentiated with respect to time. Differentiating Eq. (A.52) once with respect to time,

$$P_r (\Phi_q)' P_c X_0 + P_r \Phi_q P_c {X_0}' = P_r (\Phi_q)' P_c X_0 + \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} \begin{pmatrix} \left(-{\Phi_u}^{-1}\Phi_v\right)' \\ 0 \end{pmatrix} = 0 \tag{A.53}$$

Consequently,

$$\left(-{\Phi_u}^{-1}\Phi_v\right)' = -{\Phi_u}^{-1} P_r (\Phi_q)' P_c X_0 \tag{A.54}$$

As a result,

$$
\begin{aligned}
{X_0}' &= \begin{pmatrix} (-{\Phi_u}^{-1}\Phi_v)' \\ 0 \end{pmatrix} = \begin{pmatrix} -{\Phi_u}^{-1}P_r(\Phi_q)'P_cX_0 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} -{\Phi_u}^{-1}P_r(\Phi_q)'P_c \\ 0 \end{pmatrix} X_0 = X_2X_0 \qquad \text{(A.55)}
\end{aligned}
$$

where

$$
X_2 \equiv \begin{pmatrix} -{\Phi_u}^{-1}P_r(\Phi_q)'P_c \\ 0 \end{pmatrix} = \begin{pmatrix} -{\Phi_u}^{-1} \\ 0 \end{pmatrix} P_r(\Phi_q)'P_c \qquad \text{(A.56)}
$$

Using matrix X_1 defined by Eq. (A.51), matrix X_2 is re-written as

$$
X_2 = -X_1P_r(\Phi_q)'P_c \qquad \text{(A.57)}
$$

Differentiating Eq. (A.53) once with respect to time,

$$
\begin{aligned}
P_r(\Phi_q)''P_cX_0 &+ 2P_r(\Phi_q)'P_c{X_0}' + P_r\Phi_qP_c{X_0}'' \\
&= P_r(\Phi_q)''P_cX_0 + 2P_r(\Phi_q)'P_c{X_0}' \\
&+ \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} \begin{pmatrix} (-{\Phi_u}^{-1}\Phi_v)'' \\ 0 \end{pmatrix} = 0 \qquad \text{(A.58)}
\end{aligned}
$$

Using matrix X_2 defined by Eq. (A.57),

$$
(-{\Phi_u}^{-1}\Phi_v)'' = -{\Phi_u}^{-1}\big(2P_r(\Phi_q)'P_cX_2X_0 + P_r(\Phi_q)''P_cX_0\big) \qquad \text{(A.59)}
$$

As a result,

$$
\begin{aligned}
{X_0}'' &= \begin{pmatrix} (-{\Phi_u}^{-1}\Phi_v)'' \\ 0 \end{pmatrix} = \begin{pmatrix} -{\Phi_u}^{-1}\big(2P_r(\Phi_q)'P_cX_2X_0 + P_r(\Phi_q)''P_cX_0\big) \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} -{\Phi_u}^{-1}\big(2P_r(\Phi_q)'P_cX_2 + P_r(\Phi_q)''P_c\big) \\ 0 \end{pmatrix} X_0 = X_3X_0 \qquad \text{(A.60)}
\end{aligned}
$$

where

$$\begin{aligned} X_3 &\equiv \begin{pmatrix} -{\Phi_u}^{-1}\big(2P_r(\Phi_q)'P_cX_2 + P_r(\Phi_q)''P_c\big) \\ 0 \end{pmatrix} \\ &= \begin{pmatrix} -{\Phi_u}^{-1} \\ 0 \end{pmatrix}\big(2P_r(\Phi_q)'P_cX_2 + P_r(\Phi_q)''P_c\big) \\ &= -X_1\big(2P_r(\Phi_q)'P_cX_2 + P_r(\Phi_q)''P_c\big) \end{aligned} \tag{A.61}$$

Substituting for X_2 of Eq. (A.57) in Eq. (A.61)

$$X_3 = 2X_1P_r(\Phi_q)'P_cX_1P_r(\Phi_q)'P_c - X_1P_r(\Phi_q)''P_c \tag{A.62}$$

In order to compute the first derivative ${X_1}'$ of matrix X_1 defined by Eq. (A.51), the identity

$$P_r\Phi_qP_cX_1 = \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix}\begin{pmatrix} {\Phi_u}^{-1} \\ 0 \end{pmatrix} = I_m \tag{A.63}$$

in which I_m is the $m \times m$ identity matrix, is differentiated once with respect to time,

$$P_r(\Phi_q)'P_cX_1 + P_r\Phi_qP_c{X_1}' = P_r(\Phi_q)'P_cX_1 + \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix}\begin{pmatrix} ({\Phi_u}^{-1})' \\ 0 \end{pmatrix} = 0 \tag{A.64}$$

Consequently,

$$({\Phi_u}^{-1})' = -{\Phi_u}^{-1}P_r(\Phi_q)'P_cX_1 \tag{A.65}$$

As a result,

$$\begin{aligned} {X_1}' &= \begin{pmatrix} ({\Phi_u}^{-1})' \\ 0 \end{pmatrix} = \begin{pmatrix} -{\Phi_u}^{-1}P_r(\Phi_q)'P_cX_1 \\ 0 \end{pmatrix} \\ &= \begin{pmatrix} -{\Phi_u}^{-1} \\ 0 \end{pmatrix} P_r(\Phi_q)'P_cX_1 = -X_1P_r(\Phi_q)'P_cX_1 = X_2X_1 \end{aligned} \tag{A.66}$$

Differentiating Eq. (A.64) once with respect to time,

$$\begin{aligned} P_r(\Phi_q)''P_cX_1 \;&+\; 2P_r(\Phi_q)'P_cX_1' + P_r\Phi_qP_cX_1'' \\ &= P_r(\Phi_q)''P_cX_1 + 2P_r(\Phi_q)'P_cX_1' \\ &+ \begin{pmatrix} \Phi_u & \Phi_v \end{pmatrix} \begin{pmatrix} (\Phi_u{}^{-1})'' \\ 0 \end{pmatrix} = 0 \end{aligned} \tag{A.67}$$

Consequently,

$$(\Phi_u{}^{-1})'' = -\Phi_u{}^{-1}\big(2P_r(\Phi_q)'P_c(-X_1P_r(\Phi_q)'P_cX_1) + P_r(\Phi_q)''P_cX_1\big) \tag{A.68}$$

As a result,

$$\begin{aligned} X_1'' \;&=\; \begin{pmatrix} (\Phi_u{}^{-1})'' \\ 0 \end{pmatrix} = \begin{pmatrix} -\Phi_u{}^{-1}\big(2P_r(\Phi_q)'P_c(-X_1P_r(\Phi_q)'P_cX_1) + P_r(\Phi_q)''P_cX_1\big) \\ 0 \end{pmatrix} \\ &= 2\begin{pmatrix} \Phi_u{}^{-1} \\ 0 \end{pmatrix} P_r(\Phi_q)'P_cX_1P_r(\Phi_q)'P_cX_1 + \begin{pmatrix} -\Phi_u{}^{-1} \\ 0 \end{pmatrix} P_r(\Phi_q)''P_cX_1 \\ &= \big(-X_1P_r(\Phi_q)''P_c + 2X_1P_r(\Phi_q)'P_cX_1P_r(\Phi_q)'P_c\big)X_1 = X_3X_1 \end{aligned} \tag{A.69}$$

A.3 Boundedness of Matrix S

Stability results of Theorems 3.3 and 3.6 rely on boundedness of matrix S, constructed such that it meets the properties of Eqs. (3.11) and (3.12).

Theorem A.3. *Matrix S, having the properties of Eqs.* (3.11) *and* (3.12)*, is bounded.*

Proof. Matrix $\begin{pmatrix} R^\top & C^\top \end{pmatrix}_{n\times n}$ is nonsingular [5], where R has the properties of Eqs. (3.6) through (3.9), $\|R\| > k_R$, $\kappa(RR^\top) < K_{RR^\top} < \infty$, and $C = -\Phi_q$.

Therefore, the linearly independent columns $r_1, \ldots, r_d, c_1, \ldots, c_m$, of matrices

$$R^\top \equiv \begin{pmatrix} r_1 & \ldots & r_d \end{pmatrix}$$

and

$$C^\top \equiv \begin{pmatrix} c_1 & \ldots & c_m \end{pmatrix}$$

form a basis of $\mathbb{R}^n$. Since $\mathcal{B}_n \equiv \{r_1, \ldots, r_d, c_1, \ldots, c_m\}$ is a basis, any vector $\xi \in \mathbb{R}^n$ is a linear combination of vectors in the basis,

$$\xi = \sum_{i=1}^{d} \alpha_i^R r_i + \sum_{j=1}^{m} \alpha_j^C c_j = R^\top \alpha^R + C^\top \alpha^C$$

in which the $d \times 1$ vector α^R is defined as $\alpha^R \equiv \begin{pmatrix} \alpha_1^R & \ldots & \alpha_d^R \end{pmatrix}^\top$, and the $m \times 1$ vector α^C is defined as $\alpha^C \equiv \begin{pmatrix} \alpha_1^C & \ldots & \alpha_m^C \end{pmatrix}^\top$. As a result, each of the d column vectors $s_i, i = 1, 2, \ldots, d$, of matrix $S = \begin{pmatrix} s_1 & \ldots & s_d \end{pmatrix}$ are linear combinations of vectors in the basis,

$$s_i = R^\top \gamma_i^R + C^\top \gamma_i^C$$

Therefore, matrix S is written as

$$S = R^\top \Gamma^R + C^\top \Gamma^C \tag{A.70}$$

where $\Gamma^R \equiv \begin{pmatrix} \gamma_1^R & \ldots & \gamma_d^R \end{pmatrix}$ and $\Gamma^C \equiv \begin{pmatrix} \gamma_1^C & \ldots & \gamma_d^C \end{pmatrix}$. Pre-multiplying Eq. (A.70) by matrix R, and accounting for the fact that $RC^\top = RB = 0$, the following identity is obtained:

$$I_d = RS = RR^\top \Gamma^R$$

Matrix $RR^\top$ is positive-definite, due to the property of Eq. (3.8) that $rank(R) = d$. Therefore,

$$\Gamma^R = (RR^\top)^{-1} \tag{A.71}$$

Pre-multiplying Eq. (A.70) by matrix C and accounting for the fact that $CR^\top = (RB)^\top = 0$, the following identity is obtained:

$$0 = CS = CC^\top \Gamma^C$$

Matrix $= CC^\top$ is positive definite, because C has full row-rank. Therefore,

$$\Gamma^C = 0 \tag{A.72}$$

As a result of the Eqs. (A.71) and (A.72), matrix S is re-written as

$$S = R^\top (RR^\top)^{-1}$$

Therefore,

$$\|S\| \leq \|R^\top\| \|(RR^\top)^{-1}\| \leq \|R^\top\| \frac{K_{RR^\top}}{\|RR^\top\|} \leq K \frac{K_{RR^\top}}{{k_R}^2}$$

■

A.4 Definitions of Orientation Matrices

Consider $p \in \mathbb{R}^4$ the vector of Euler parameters of a body; i.e., a four-dimensional normalized vector, $\|p\|_2 = 1$,

$$p = \begin{pmatrix} e_0 & e_1 & e_2 & e_3 \end{pmatrix}^\top = \begin{pmatrix} e_0 & e^\top \end{pmatrix}^\top \tag{A.73}$$

where $e = \begin{pmatrix} e_1 & e_2 & e_3 \end{pmatrix}^\top$. The orientation matrix of a body is uniquely defined by its Euler parameter vector p [27],

$$\begin{aligned} A(p) &\equiv (e_0^2 - e^\top e)I + 2ee^\top + 2e_0\tilde{e} \\ &= \begin{pmatrix} e_0^2 + e_1^2 - e_2^2 - e_3^2 & 2(e_1e_2 - e_0e_3) & 2(e_1e_3 + e_0e_2) \\ 2(e_1e_2 + e_0e_3) & e_0^2 - e_1^2 + e_2^2 - e_3^2 & 2(e_2e_3 - e_0e_1) \\ 2(e_1e_3 - e_0e_2) & 2(e_2e_3 + e_0e_1) & e_0^2 - e_1^2 - e_2^2 + e_3^2 \end{pmatrix} \end{aligned} \tag{A.74}$$

Matrix $A(p)$ is orthogonal ; i.e.,

$$AA^\top = A^\top A = (p^\top p)^2 I = I \tag{A.75}$$

Other matrices depending on p, which are used to define the orientation matrix$A(p)$ and kinematic terms; e.g., the mass matrix M_i and the Coriolis vector L_i of a body i, are matrices $E(p)$ and $G(p)$ [27],

$$E(p) \equiv \begin{pmatrix} -e & \tilde{e} + e_0 I \end{pmatrix} \tag{A.76}$$

$$G(p) \equiv \begin{pmatrix} -e & -\tilde{e} + e_0 I \end{pmatrix} \tag{A.77}$$

with properties [27]

$$Ep = 0 \tag{A.78}$$

$$Gp = 0 \tag{A.79}$$

The following identities hold between the orientation matrices $A(p)$, $E(p)$, and $G(p)$ [27]:

$$A = EG^\top \tag{A.80}$$

$$EE^\top = GG^\top = p^\top pI = I \tag{A.81}$$

$$E^\top E = G^\top G = p^\top pI - pp^\top = I - pp^\top \tag{A.82}$$

A.5 Partial Derivatives of Orientation Matrices

Consider vectors $a, a', b \in \mathbb{R}^3$ and $\gamma \in \mathbb{R}^4$, not normalized. The following identities hold [50]:

$$\big(E(p)\gamma\big)_p = -E(\gamma) \tag{A.83}$$

$$\big(E^\top(p)a\big)_p = (a)^+ \tag{A.84}$$

where matrix $(a)^+$ is defined [43] as

$$(a)^+ \equiv \begin{pmatrix} 0 & -a^\top \\ a & \tilde{a} \end{pmatrix} \tag{A.85}$$

and has the following properties:

$$(a)^+ p = E^\top(p)a \tag{A.86}$$

$$\big(G(p)\gamma\big)_p = -G(\gamma) \tag{A.87}$$

$$\big(G^\top(p)a\big)_p = (a)^- \tag{A.88}$$

where matrix $(a)^-$ is defined [43] as

$$(a)^- \equiv \begin{pmatrix} 0 & -a^\top \\ a & -\tilde{a} \end{pmatrix} \tag{A.89}$$

and has the property

$$(a)^- p = G^\top(p)a \tag{A.90}$$

The derivative of the product $(A(p)a')$ with respect to vector p is [50]

$$B(p,a')\equiv\big(A(p)a'\big)_p = 2\begin{pmatrix} (e_0 I+\tilde{e})a' & ea'^{\top}-(e_0 I+\tilde{e})\tilde{a}' \end{pmatrix} \tag{A.91}$$

where the 3×4 matrix $B(p,a')$ has the following properties [50]:

$$B(p_i,a')p_j = B(p_j,a')p_i \tag{A.92}$$

$$\big(B(p_i,a')p_j\big)_{p_i} = B(p_j,a') \tag{A.93}$$

$$\big(B(p_i,a')p_j\big)_{p_i} = B(p_j,a') \tag{A.94}$$

The derivative of the product $(A^{\top}(p)a')$ with respect to vector p is [50]

$$C(p,a)\equiv\big(A^{\top}(p)a\big)_p = 2\begin{pmatrix} (e_0 I-\tilde{e})a & ea^{\top}+(e_0 I-\tilde{e})\tilde{a} \end{pmatrix} \tag{A.95}$$

where the 3×4 matrix $C(p,a)$ has the following properties [50]:

$$C(p_i,a)p_j = C(p_j,a)p_i \tag{A.96}$$

$$\big(C(p_i,a)p_j\big)_{p_i} = C(p_j,a) \tag{A.97}$$

The derivative of the product $(B(p_i,a'_i)p_j)$ with respect to vector a' is [50]

$$N(p_i,p_j)\equiv\big(B(p_i,a'_i)p_j\big)_{a'_i} = 2\{E(p_i)G^{\top}(p_j)\} \tag{A.98}$$

where the 3×3 matrix $N(p_i,p_j)$ has the following properties [50]:

$$N^{\top}(p_i,p_j) = \big(C(p_i,a_i)p_j\big)_{a_i} \tag{A.99}$$

$$N(p_i,p_j) = N(p_j,p_i) \tag{A.100}$$

The derivative of the product $B^\top(p,a')b$ with respect to vector a' is [50]

$$\big(B^\top(p,a')b\big)_{a'} = C^\top(p,b) \tag{A.101}$$

and the derivative of the product $C^\top(p,b)a'$ with respect to vector b is [50]

$$\big(C^\top(p,b)a'\big)_b = B^\top(p,a') \tag{A.102}$$

A.6 Equations of Motion in a Non-Centroidal Coordinate Frame

The centroid of a body is located [27], relative to a body-fixed coordinate frame,

$$\tilde{s'}^C = \frac{1}{m}\int_m s'^P dm(P) \tag{A.103}$$

where m is the mass of the body and $dm(P)$ is the differential mass element located at point P. The inertia tensor is defined [27] as

$$J' = -\int_m \tilde{s'}^P \tilde{s'}^P dm(P) \tag{A.104}$$

The variational form of the equations of motions is [27]

$$\begin{aligned} &\delta r^\top\big(mr'' + m(A\widetilde{\omega'} + A\tilde{\omega}\tilde{\omega}) - F^A\big) \\ +\quad &\delta\pi^\top\big(m\tilde{s'}^C A^\top r'' + J'\omega' + \widetilde{\omega}J'\omega\tilde{\omega} - n'^A\big) = 0 \end{aligned} \tag{A.105}$$

where [27] δr is the virtual displacement of the origin of the body-fixed coordinate frame, $\delta\pi$ is the virtual rotation of the body, ω is the angular velocity of the body-fixed coordinate frame,

$$F^A = \int_m F(P)dm(P) \tag{A.106}$$

is the applied force acting on the body, and

$$n'^{A} = \int_m \tilde{s}'^{P} F'(P) dm(P) \tag{A.107}$$

is the applied torque resulting from the applied force acting on the body. The force-per-unit of mass $F(P)$ is a function of the position of a point P inside the body, expressed in the global coordinate frame. The force-per-unit of mass $F'(P)$ and the applied torque n'^{A} are expressed in the body-fixed coordinate frame. Hence,

$$F(P) = AF'(P) \tag{A.108}$$

and, pre-multiplying Eq. (A.108) by $A^{-1} = A^{\top}$,

$$F'(P) = A^{\top} F(P) \tag{A.109}$$

Substituting for $F'(P)$ in Eq. (A.107), the applied torque is re-written as

$$n'^{A} = \int_m \tilde{s}'^{P} A^{\top} F(P) dm(P) \tag{A.110}$$

It is to be noted that the applied torque depends on the applied force through Eq. (A.110); e.g., for a constant force-per-unit of mass

$$F(P) = \frac{F}{m} \tag{A.111}$$

the applied torque is

$$n'^{A} = \int_m \tilde{s}'^{P} A^{\top} \frac{F}{m} dm(P) = \frac{1}{m} \Big(\int_m \tilde{s}'^{P} \Big) A^{\top} F \tag{A.112}$$

Using the definition of the centroid, of Eq. (A.103), the applied torque resulting from a constant force-per-unit of mass is

$$n'^{A} = \frac{1}{m} \Big(\int_m \tilde{s}'^{P} \Big) A^{\top} F = \frac{1}{m} m \tilde{s}'^{C} A^{\top} F = \tilde{s}'^{C} A^{\top} F \tag{A.113}$$

Using the following identities [27]:

$$\omega' = 2\delta p G p''$$

$$\delta\pi = 2G\delta p$$

$$\omega = 2Gp'$$

$$\tilde{\omega} = A^\top A' = -A'^\top A$$

and

$$\tilde{\omega} s'^C = -\tilde{s}'^C \omega$$

the variational equation of motion is re-written as

$$\begin{aligned} & \delta r^\top \left(m r'' - 2mA\tilde{s}'^C G p'' - mAA^\top A' A'^\top A s'^C - F^A \right) \\ + \quad & 2\delta p^\top G^\top \left(m\tilde{s}'^C A^\top r'' + 2J'Gp'' + 2A^\top A' J' G p' - n'^A \right) = 0 \end{aligned} \tag{A.114}$$

Since [27] $A = EG^\top$, $A' = 2EG'^\top$, and $E^\top E = I - pp^\top$,

$$A^\top A' = GE^\top 2EG'^\top = 2G(I - pp^\top)G'^\top$$

Using the identity [27] $Gp = 0$, it follows that

$$A^\top A' = 2GG'^\top$$

Substituting for $A^\top A'$ in Eq. (A.114)

$$\begin{aligned} & \delta r^\top \left(m r'' - 2mA\tilde{s}'^C G p'' - mAA^\top A' A'^\top A s'^C - F^A \right) \\ + \quad & \delta p^\top \left(2mG^\top \tilde{s}'^C A^\top r'' + 4G^\top J' G p'' + 8G^\top G G'^\top J' G p' - 2G^\top n'^A \right) = 0 \end{aligned} \tag{A.115}$$

and using the identities [27] $G^\top G = I - pp^\top$ and $A' = 2EG'^\top$, the variational equation of motion is re-written as

$$\delta q^\top \left(\left(\begin{array}{cc} mI & -2mA\tilde{s'}^C G \\ 2mG^\top \tilde{s'}^C A^\top & 4G^\top J'G \end{array} \right) q'' - \left(\begin{array}{c} 4mEG'^\top G'G^\top s'^C \\ -8G'^\top J'Gp' \end{array} \right) \right)$$
$$- \quad \delta q^\top \left(\left(\begin{array}{c} 0 \\ 8pp^\top G'^\top J'Gp' \end{array} \right) + \left(\begin{array}{c} F^A \\ 2G^\top n'^A \end{array} \right) \right) = 0 \tag{A.116}$$

Defining

$$M_0 \equiv \left(\begin{array}{cc} mI & -2mA\tilde{s'}^C G \\ 2mG^\top \tilde{s'}^C A^\top & 4G^\top J'G \end{array} \right) \tag{A.117}$$

$$L_0 \equiv \left(\begin{array}{c} 4mEG'^\top G'G^\top s'^C \\ -8G'^\top J'Gp' \end{array} \right) \tag{A.118}$$

$$L_{0,1} \equiv \left(\begin{array}{c} 0 \\ 8pp^\top G'^\top J'Gp' \end{array} \right) \tag{A.119}$$

$$Q_0 \equiv \left(\begin{array}{c} F^A \\ 2G^\top n'^A \end{array} \right) \tag{A.120}$$

and noting that

$$\delta q^\top L_{0,1} = 0$$

since [27] $\delta p^\top p = 0$, the variational equation of motion reduces to

$$\delta q^\top \Big(M_0 q'' - L_0 - Q_0 \Big) = 0 \tag{A.121}$$

This equation must hold for all kinetically admissible δq; i.e., for all δq such that

$$\Phi_q \delta q^\top = 0 \tag{A.122}$$

There must exist [27] a Lagrange multiplier vector λ such that

$$\delta q^{\top}\Big(M_0 q'' - L_0 - Q_0 + {\Phi_q}^{\top}\lambda\Big) = 0 \tag{A.123}$$

for an arbitrary virtual displacement vector δq. Thus, its coefficient must be zero, yielding

$$M_0 q'' + {\Phi_q}^{\top}\lambda = S_0^1 \tag{A.124}$$

where $S_0^1 = L_0 + Q_0$.

For a multibody system comprising n_b bodies, the equations of motion for each body have the same form,

$$M_i {q_i}'' + {\Phi_{q_i}}^{\top}\lambda = S_i^1 \tag{A.125}$$

where

$$M_i \equiv \begin{pmatrix} m_i I & -2m_i A_i \tilde{s'}_i^C G_i \\ 2m_i {G_i}^{\top} \tilde{s'}_i^C {A_i}^{\top} & 4{G_i}^{\top} {J'}_i G_i \end{pmatrix} \tag{A.126}$$

$$S_i^1 = L_i + Q_i \tag{A.127}$$

$$L_i \equiv \begin{pmatrix} 4m_i E_i {G_i'}^{\top} G_i' {G_i}^{\top} {s'}_i^C \\ -8{G_i'}^{\top} {J'}_i G_i {p_i}' \end{pmatrix} \tag{A.128}$$

and

$$Q_i \equiv \begin{pmatrix} F_i^A \\ 2{G_i}^{\top} {n'}_i^A \end{pmatrix} \tag{A.129}$$

in which m_i is the i-th body mass; ${J'}_i$ is the i-th body inertia tensor; ${s'}_i^C$ is the i-th body position of the center of mass with respect to its body-fixed coordinate frame; $q_i = \begin{pmatrix} {r_i}^{\top} & {p_i}^{\top} \end{pmatrix}^{\top}$, r_i being the position of the i-th body body-fixed coordinate frame origin with respect to global coordinate frame and p_i being the Euler parameters

vector of the i-th body; $A_i \equiv A(p_i)$, $E_i \equiv E(p_i)$, and $G_i \equiv G(p_i)$; F_i^A is the applied force acting on body i; and ${n'}_i^A$ is the applied torque acting on body i, resulting form the applied force F_i^A. Defining

$$q \equiv \begin{pmatrix} {q_1}^\top & \dots & {q_{n_b}}^\top \end{pmatrix}^\top \tag{A.130}$$

$$M \equiv diag(M_i, i = 1, 2, \dots, n_b) \tag{A.131}$$

$$L \equiv \begin{pmatrix} {L_1}^\top & \dots & {L_{n_b}}^\top \end{pmatrix}^\top \tag{A.132}$$

$$Q \equiv \begin{pmatrix} {Q_1}^\top & \dots & {Q_{n_b}}^\top \end{pmatrix}^\top \tag{A.133}$$

and $S^1 = L + Q$, the multibody system equation of motion, obtained by collecting equations of the form of Eq. (A.125) and using the definitions of Eqs. (A.130) through (A.133), is

$$Mq'' + {\Phi_q}^\top \lambda = S^1 \tag{A.134}$$

This equation of motion and the constraint equation

$$\Phi(q, \beta, t) = 0 \tag{A.135}$$

represent the non-centroidal formulation of the equations of motion of a multibody system. They constitute an index-3 DAE [48, 27].

A.7 Partial Derivatives of Mass Matrix and Coriolis Blocks

As shown in the previous Section, the body i-th 7×7 mass matrix block M_i is

$$M_i = \begin{pmatrix} m_i I & -2m_i A(p_i)\tilde{s'}_i^C G(p_i) \\ 2m_i G(p_i)^\top \tilde{s'}_i^C A(p_i)^\top & 4G(p_i)^\top J'_i G(p_i) \end{pmatrix}$$

The product of M_i and a seven-dimensional constant vector

$$\gamma^i = \begin{pmatrix} \gamma^{i,r\top} & \gamma^{i,p\top} \end{pmatrix}^\top$$

is

$$M_i\gamma^i = \begin{pmatrix} m_i\gamma^{i,r} - 2m_i A(p_i)\tilde{s'}_i^C G(p_i)\gamma^{i,p} \\ 2m_i G(p_i)^\top \tilde{s'}_i^C A(p_i)^\top \gamma^{i,r} + 4G(p_i)^\top J'_i G(p_i)\gamma^{i,p} \end{pmatrix} \tag{A.136}$$

The product $M_i\gamma^i$ depends only on the body i Euler parameters vector p_i. Therefore,

$$(M_i\gamma^i)_{r_i} = \begin{pmatrix} 0 \\ \\ 0 \end{pmatrix} \tag{A.137}$$

The partial derivative of the upper-block of Eq. (A.136) with respect to p_i is

$$M_{1,i} = m_i(\gamma^{i,r})_{p_i} - 2m_i(A(p_i)\tilde{s'}_i^C G(p_i)\gamma^{i,p})_{p_i} \tag{A.138}$$

where the fact that the mass of body i does not depend on p_i is accounted for. Since γ^i is assumed to be a constant vector

$$m_i(\gamma^{i,r})_{p_i} = 0 \tag{A.139}$$

Using the product and chain rules of differentiation,

$$(A(p_i)\tilde{s'}_i^C G(p_i)\gamma^{i,p})_{p_i} = (A(p_i)(\tilde{s'}_i^C \widehat{G(p_i)\gamma^{i,p}}))_{p_i} + A(p_i)\tilde{s'}_i^C (G(p_i)\gamma^{i,p})_{p_i} \quad \text{(A.140)}$$

and using the differentiation identities of Eqs. (A.91) and (A.87),

$$(A(p_i)\tilde{s'}_i^C G(p_i)\gamma^{i,p})_{p_i} = B(p_i, \tilde{s'}_i^C G_i\gamma^{i,p}) - A_i\tilde{s'}_i^C G(\gamma^{i,p}) \quad \text{(A.141)}$$

Substituting expressions of Eqs. (A.139) through (A.141) in Eq. (A.138),

$$M_{1,i} = -2m_i B(p_i, \tilde{s'}_i^C G_i\gamma^{i,p}) + 2m_i A_i \tilde{s'}_i^C G(\gamma^{i,p}) \quad \text{(A.142)}$$

The partial derivative of the lower-block of Eq. (A.136) with respect to p_i is

$$M_{2,i} = 2m_i(G(p_i)^\top \tilde{s'}_i^C A(p_i)^\top \gamma^{i,r})_{p_i} + 4(G(p_i)^\top J'_i G(p_i)\gamma^{i,p})_{p_i} \quad \text{(A.143)}$$

Using the product and chain rules of differentiation,

$$\begin{aligned} M_{2,i} &= 2m_i(G(p_i)^\top(\tilde{s'}_i^C \widehat{A(p_i)^\top \gamma^{i,r}}))_{p_i} \\ &+ 2m_i G(p_i)^\top \tilde{s'}_i^C (A(p_i)^\top \gamma^{i,r})_{p_i} \\ &+ 4(G(p_i)^\top(J'_i \widehat{G(p_i)\gamma^{i,p}}))_{p_i} \\ &+ 4G(p_i)^\top J'_i (G(p_i)\gamma^{i,p})_{p_i} \end{aligned} \quad \text{(A.144)}$$

and using the differentiation identities of Eqs. (A.88), (A.95) and (A.87),

$$M_{2,i} = 2m_i(\tilde{s'}_i^C A_i{}^\top \gamma^{i,r})^- + 2m_i G_i{}^\top \tilde{s'}_i^C C(p_i, \gamma^{i,r}) + 4(J'_i G_i \gamma^{i,p})^- - 4G_i{}^\top J'_i G(\gamma^{i,p}) \quad \text{(A.145)}$$

As a result, matrix $(M_i\gamma^i)_{q_i}$ is

$$(M_i\gamma^i)_{q_i} = \begin{pmatrix} 0 & M_{1,i} \\ 0 & M_{2,i} \end{pmatrix} \quad \text{(A.146)}$$

The partial derivative $((M_i {\gamma_1}^i)_{q_i} {\gamma_2}^i)_{q_i}$, where ${\gamma_1}^i$ and ${\gamma_2}^i$ are constant 7-dimensional vectors, is

$$((M_i {\gamma_1}^i)_{q_i} {\gamma_2}^i)_{q_i} = \begin{pmatrix} 0 & M_{3,i} \\ 0 & M_{4,i} \end{pmatrix} \tag{A.147}$$

where

$$M_{3,i} = -2mB(\gamma_2^{i,p}, \tilde{s}'^C G {\gamma_1}^{i,p}) + 2mN(p_i, \gamma_2^{i,p})\tilde{s}'^C G(\gamma_1^{i,p}) + 2mB(p_i, \tilde{s}'^C G(\gamma_1^{i,p})\gamma_2^{i,p})$$

and

$$\begin{aligned} M_{4,i} &= 2mG^\top(\gamma_2^{i,p})\tilde{s}'^C C(p_i, {\gamma_1}^{i,r}) + 2m(\tilde{s}'^C C(p_i, {\gamma_1}^{i,r})\gamma_2^{i,p})^- \\ &+ 2mG^\top \tilde{s}'^C C(\gamma_2^{i,p}, {\gamma_1}^{i,r}) - 4G^\top(\gamma_2^{i,p})J'G(\gamma_1^{i,p}) - 4(J'G(\gamma_1^{i,p})\gamma_2^{i,p})^- \end{aligned}$$

The derivatives of Coriolis term L_i, defined in Eq. (A.128), with respect to r_i and ${r_i}'$ are zero, since L_i depends only on p_i and ${p_i}'$. The derivative of L_i with respect to p_i is

$$L_{ip_i} = \begin{pmatrix} 4m_i(E(p_i)G({p_i}')^\top G({p_i}')G^\top(p_i){s'}_i^C)_{p_i} \\ -8(G({p_i}')^\top {J'}_i G(p_i){p_i}')_{p_i} \end{pmatrix} \tag{A.148}$$

The upper-block of Eq. (A.148) is

$$\begin{aligned} L_{i1,p_i} &= 4m_i(E(p_i)(G({p_i}')^\top \widehat{G({p_i}')G}^\top(p_i){s'}_i^C))_{p_i} \\ &+ 4m_i E(p_i)G({p_i}')^\top G({p_i}')(G^\top(p_i){s'}_i^C)_{p_i} \end{aligned} \tag{A.149}$$

Using the identities of Eqs. (A.83) and (A.88),

$$\begin{aligned} L_{i1,p_i} &= -4m_i E(G({p_i}')^\top G({p_i}')G(p_i)^\top {s'}_i^C) \\ &+ 4m_i E(p_i)G({p_i}')^\top G({p_i}')({s'}_i^C)^- \end{aligned} \tag{A.150}$$

The lower-block of Eq. (A.148) is

$$L_{i2,p_i} = -8G(p_i')^\top J'_i(G(p_i)p_i')_{p_i} \tag{A.151}$$

Using the identity of Eq. (A.87),

$$L_{i2,p_i} = 8G(p_i')^\top J'_i G(p_i') \tag{A.152}$$

As a result, matrix L_{iq_i} is

$$L_{iq_i} = \begin{pmatrix} 0 & L_{i1,p_i} \\ 0 & L_{i2,p_i} \end{pmatrix} \tag{A.153}$$

The derivative of L_i with respect to p_i' is

$$L_{ip_i'} = \begin{pmatrix} 4m_i(E(p_i)G(p_i')^\top G(p_i')G^\top(p_i)s'^C_i)_{p_i'} \\ -8(G(p_i')^\top J'_i G(p_i)p_i')_{p_i'} \end{pmatrix} \tag{A.154}$$

The upper-block of Eq. (A.154) is

$$\begin{aligned} L_{i1,p_i'} &= 4m_i E(p_i)(G(p_i')^\top(G(p_i')\widehat{G^\top(p_i)}s'^C_i))_{p_i'} \\ &+ 4m_i E(p_i)G(p_i')^\top(G(p_i')(G^\top(p_i)s'^C_i))_{p_i'} \end{aligned} \tag{A.155}$$

Using the identities of Eqs. (A.88) and (A.87),

$$\begin{aligned} L_{i1,p_i'} &= 4m_i E(p_i)((G(p_i')G^\top(p_i)s'^C_i))^- \\ &- 4m_i E(p_i)G(p_i')^\top G((G^\top(p_i)s'^C_i)) \end{aligned} \tag{A.156}$$

The lower-block of Eq. (A.154) is

$$\begin{aligned} L_{i2,p_i'} &= -8(G(p_i')^\top(J'_i\widehat{G(p_i)}p_i'))_{p_i'} \\ &- 8G(p_i')^\top J'_i G(p_i) \end{aligned} \tag{A.157}$$

Using the identity of Eq. (A.88),

$$L_{i2,p_i'} = -8(J'_i G(p_i) p_i')^- - 8G(p_i')^\top J'_i G(p_i) \tag{A.158}$$

As a result, matrix $L_{iq_i'}$ is

$$L_{iq_i'} = \begin{pmatrix} 0 & L_{i1,p_i'} \\ 0 & L_{i2,p_i'} \end{pmatrix} \tag{A.159}$$

Partial derivative $(L_{iq_i'}\gamma^i)_{q_i}$ is

$$(L_{iq_i'}\gamma^i)_{q_i} = \begin{pmatrix} 0 & L_{0,i} \\ 0 & 8G^\top(\gamma^{i,p})J'G(p_i') + 8(G(p_i'))^\top J'G(\gamma^{i,p}) \end{pmatrix} \tag{A.160}$$

where

$$\begin{aligned} L_{0,i} = & - 4mE(G^\top(\gamma^{i,p})G(p_i')G^\top s'^C) + 4mEG^\top(\gamma^{i,p})G(p_i')(s'^C)^- \\ & + 4mE((G(p_i'))^\top G(G^\top s'^C)\gamma^{i,p}) + 4mE(G(p_i'))^\top G(\gamma^{i,p})(s'^C)^- \end{aligned}$$

and partial derivative $(L_{iq_i'}\gamma^i)_{q_i'}$ is

$$(L_{iq_i'}\gamma^i)_{q_i'} = \begin{pmatrix} 0 & -4mEG^\top(\gamma^{i,p})G(G^\top s'^C) - 4mE(G(G^\top s'^C)\gamma^{i,p})^- \\ 0 & -8G^\top(\gamma^{i,p})J'G - 8(J'G\gamma^{i,p})^- \end{pmatrix} \tag{A.161}$$

A.8 Partial Derivatives of Force Terms

Generalized forces Q_i and Q_j acting on bodies i and j, have the form [50]

$$Q_i = \begin{pmatrix} F_i^A \\ 2{G_i}^\top {n_i'}^A \end{pmatrix} + Q_i^{TSDA} + Q_i^{RSDA} \tag{A.162}$$

$$Q_j = \begin{pmatrix} F_j^A \\ 2{G_j}^\top {n_j'}^A \end{pmatrix} + Q_j^{TSDA} + Q_j^{RSDA} \tag{A.163}$$

Kinematic derivatives of

$$Q_i^A = \begin{pmatrix} F_i^A \\ 2{G_i}^\top {n_i'}^A \end{pmatrix} \tag{A.164}$$

are

$$(Q_i^A)_{r_i} = 0 \tag{A.165}$$

$$(Q_i^A)_{p_i} = \begin{pmatrix} 0 \\ 2(\tilde{s'^C} {A_i}^\top F_i^A)^- + 2{G_i}^\top \tilde{s'^C} C(p_i, F_i^A) \end{pmatrix} \tag{A.166}$$

$$(Q_i^A)_{q_{i'}} = 0 \tag{A.167}$$

The TSDA forces have the form [50]

$$Q_i^{TSDA} = \frac{f}{l} \begin{pmatrix} d_{ij} \\ (B^\top(p_i, {s'}_i^p) d_{ij}) \end{pmatrix} \tag{A.168}$$

$$Q_j^{TSDA} = -\frac{f}{l} \begin{pmatrix} d_{ij} \\ (B^\top(p_j, {s'}_j^p) d_{ij}) \end{pmatrix} \tag{A.169}$$

where

$$f = k(l - l_0) + cl' + F(l, l') \tag{A.170}$$

$$l^2 = {d_{ij}}^\top d_{ij} \tag{A.171}$$

$$d_{ij} = r_j + A_j {s'_j}^P - r_i - A_i {s'_i}^P$$

With $q_{ij} = \begin{pmatrix} {q_i}^\top & {q_j}^\top \end{pmatrix}^\top$, kinematic derivatives of TSDA forces are [50]

$$Q_{i_{q_{ij}}}^{TSDA} = \begin{pmatrix} (\frac{f}{l} d_{ij})_{q_{ij}} \\ B^\top(p_i, {s'}_i^p)(\frac{f}{l} d_{ij})_{q_{ij}} + \frac{f}{l} R_{k=1,\ldots,4}(\begin{pmatrix} 0 & {d_{ij}}^\top B(e_k, {s'}_i^p) & 0 & 0 \end{pmatrix}) \end{pmatrix} \tag{A.172}$$

$$Q^{TSDA}_{j_{q_{ij}}} = \begin{pmatrix} -(\frac{f}{l}d_{ij})_{q_{ij}} \\ -B^{\top}(p_j, s'^{p}_{j})(\frac{f}{l}d_{ij})_{q_{ij}} - \frac{f}{l}R_{k=1,\ldots,4}(\begin{pmatrix} 0 & 0 & 0 & d_{ij}{}^{\top}B(e_k, s'^{p}_{j}) \end{pmatrix}) \end{pmatrix} \tag{A.173}$$

$$Q^{TSDA}_{i_{q'_{ij}}} = \begin{pmatrix} d_{ij}(\frac{f}{l})_{q'_{ij}} \\ B^{\top}(p_i, s'^{p}_{i})d_{ij}(\frac{f}{l})_{q'_{ij}} \end{pmatrix} \tag{A.174}$$

$$Q^{TSDA}_{j_{q'_{ij}}} = \begin{pmatrix} -d_{ij}(\frac{f}{l})_{q'_{ij}} \\ -B^{\top}(p_j, s'^{p}_{j})d_{ij}(\frac{f}{l})_{q'_{ij}} \end{pmatrix} \tag{A.175}$$

$$(Q^{TSDA}_{i_{q'_{ij}}}\gamma)_{q_{ij}} = \begin{pmatrix} \left(\left(\frac{f}{l}d_{ij}\right)_{q'_{ij}}\gamma\right)_{q_{ij}} \\ B^{\top}(p_i, s'^{P}_{i})\left(\left(\frac{f}{l}d_{ij}\right)_{q'_{ij}}\gamma\right)_{q_{ij}} + R_{k=1,\ldots,4}(\left(\left(\frac{f}{l}d_{ij}\right)_{q'_{ij}}\gamma\right)^{\top})A \end{pmatrix} \tag{A.176}$$

$$(Q^{TSDA}_{j_{q'_{ij}}}\gamma)_{q_{ij}} = \begin{pmatrix} -\left(\left(\frac{f}{l}d_{ij}\right)_{q'_{ij}}\gamma\right)_{q_{ij}} \\ -B^{\top}(p_j, s'^{P}_{j})\left(\left(\frac{f}{l}d_{ij}\right)_{q'_{ij}}\gamma\right)_{q_{ij}} - R_{k=1,\ldots,4}(\left(\left(\frac{f}{l}d_{ij}\right)_{q'_{ij}}\gamma\right)^{\top})A_0 \end{pmatrix} \tag{A.177}$$

$$(Q^{TSDA}_{i_{q'_{ij}}}\gamma)_{q'_{ij}} = 0 \tag{A.178}$$

$$(Q^{TSDA}_{j_{q'_{ij}}}\gamma)_{q'_{ij}} = 0 \tag{A.179}$$

where matrix $R_{k=1,\ldots,4}({\gamma_k}^{\top})$ is defined as

$$R_{k=1,\ldots,4}({\gamma_k}^{\top}) = \begin{pmatrix} {\gamma_1}^{\top} \\ {\gamma_2}^{\top} \\ {\gamma_3}^{\top} \\ {\gamma_4}^{\top} \end{pmatrix} \tag{A.180}$$

γ, γ_1, γ_2, γ_3, and γ_4 are 14-dimensional constant vectors of the form

$$\gamma = \begin{pmatrix} {\gamma^{i,r}}^{\top} & {\gamma^{i,p}}^{\top} & {\gamma^{j,r}}^{\top} & {\gamma^{j,p}}^{\top} \end{pmatrix}^{\top} \tag{A.181}$$

and e_k is the four-dimensional unit vector with all elements zero except the k-th component, which is one. Kinematic derivatives of $\frac{f}{l}d_{ij}$ are

$$\left(\frac{f}{l}d_{ij}\right)_{q_{ij}} = \begin{pmatrix} -N_1 & -N_i & N_1 & N_j \end{pmatrix} \tag{A.182}$$

and

$$\left(\frac{f}{l}d_{ij}\right)_{{q'}_{ij}} = \frac{c}{l^2}d_{ij}{d_{ij}}^{\top}\begin{pmatrix} -I & -B(p_i, {s'}_i^P) & I & B(p_j, {s'}_j^P) \end{pmatrix} \tag{A.183}$$

$$\begin{aligned}\left(\left(\frac{f}{l}d_{ij}\right)_{{q'}_{ij}}\gamma\right)_x &= \frac{(c+F_{l'})}{l^2}\Big(\Big(-\frac{2}{l^2}d_{ij}{d_{ij}}^{\top}{d'_{ij}}_{{q'}_{ij}}\gamma{d_{ij}}^{\top} + d_{ij}({d'_{ij}}_{{q'}_{ij}}\gamma)^{\top} \\ &+ {d_{ij}}^{\top}({d'_{ij}}_{{q'}_{ij}}\gamma)I\Big){d_{ij}}_x + d_{ij}{d_{ij}}^{\top}({d'_{ij}}_{{q'}_{ij}}\gamma)_x\Big)\end{aligned} \tag{A.184}$$

where $x \in \{q, q'\}$ and

$$N_1 = \left(k - \frac{f}{l} - c\frac{l'}{l}\right)\frac{d_{ij}{d_{ij}}^{\top}}{l^2} + c\frac{d_{ij}{d'_{ij}}^{\top}}{l^2} + \frac{f}{l}I$$

$$N_2 = c\frac{d_{ij}{d_{ij}}^{\top}}{l^2}$$

$$N_i = N_1 B(p_i, {s'}_i^P) + N_2 B({p_i}', {s'}_i^P)$$

$$N_j = N_1 B(p_j, {s'}_j^P) + N_2 B({p_j}', {s'}_j^P)$$

$$d'_{ij} = r'_j + B(p_j, {s'}_j^P)p'_j - r'_i - B(p_i, {s'}_i^P)p'_i \tag{A.185}$$

$${d_{ij}}_{q_{ij}} = \begin{pmatrix} -I & -B(p_i, {s'}_i^P) & I & B(p_j, {s'}_j^P) \end{pmatrix} \tag{A.186}$$

$${d_{ij}}_{{q'}_{ij}} = 0 \tag{A.187}$$

$$({d_{ij}}'_{{q'}_{ij}}\gamma)_{q_{ij}} = \begin{pmatrix} 0 & -B(\gamma^{i,p}, {s'}_i^P) & 0 & B(\gamma^{j,p}, {s'}_j^P) \end{pmatrix} \tag{A.188}$$

$$({d_{ij}}'_{{q'}_{ij}}\gamma)_{{q'}_{ij}} = 0 \tag{A.189}$$

The RSDA forces have the form [50]

$$Q_i^{RSDA} = 2n \begin{pmatrix} 0 \\ (G_i{}^\top h_i') \end{pmatrix} \tag{A.190}$$

$$Q_j^{RSDA} = -2n \begin{pmatrix} 0 \\ (G_j{}^\top h_j') \end{pmatrix} \tag{A.191}$$

where

$$n = k_\theta(\theta + 2n_{rev}\pi) + c_\theta\theta' + N(\theta + 2n_{rev}\pi, \theta') \tag{A.192}$$

$$\cos(\theta) = f_i'{}^\top A_i{}^\top A_j f_j' \tag{A.193}$$

$$\sin(\theta) = g_i'{}^\top A_i{}^\top A_j f_j' \tag{A.194}$$

Kinematic derivatives of RSDA forces are [50]

$$Q_{i_{q_{ij}}}^{RSDA} = \begin{pmatrix} 0 \\ 2G_i{}^\top h_i' n_{q_{ij}} + 2n \begin{pmatrix} 0 & h'_i{}^- & 0 & 0 \end{pmatrix} \end{pmatrix} \tag{A.195}$$

$$Q_{j_{q_{ij}}}^{RSDA} = \begin{pmatrix} 0 \\ -2G_i{}^\top h_j' n_{q_{ij}} - 2n \begin{pmatrix} 0 & 0 & 0 & h'_j{}^- \end{pmatrix} \end{pmatrix} \tag{A.196}$$

$$Q_{i_{q'_{ij}}}^{RSDA} = \begin{pmatrix} 0 \\ 2G_i{}^\top h_i' n_{q'_{ij}} \end{pmatrix} \tag{A.197}$$

$$Q_{j_{q'_{ij}}}^{RSDA} = \begin{pmatrix} 0 \\ -2G_j{}^\top h_j' n_{q'_{ij}} \end{pmatrix} \tag{A.198}$$

$$(Q_{i_{q'_{ij}}}^{RSDA}\gamma)_{q_{ij}} = \begin{pmatrix} 0 \\ 2G_i{}^\top h_i'(n_{q'_{ij}}\gamma)_{q_{ij}} + 2n_{q'_{ij}}\gamma \begin{pmatrix} 0 & h'_i{}^- & 0 & 0 \end{pmatrix} \end{pmatrix} \tag{A.199}$$

$$(Q^{RSDA}_{j_{q'_{ij}}}\gamma)_{q_{ij}} = \begin{pmatrix} 0 \\ -2{G_j}^\top h'_j (n_{q'_{ij}}\gamma)_{q_{ij}} - 2n_{q'_{ij}}\gamma \begin{pmatrix} 0 & 0 & 0 & {h'_j}^- \end{pmatrix} \end{pmatrix} \tag{A.200}$$

$$(Q^{RSDA}_{i_{q'_{ij}}}\gamma)_{q'_{ij}} = 0 \tag{A.201}$$

$$(Q^{RSDA}_{j_{q'_{ij}}}\gamma)_{q'_{ij}} = 0 \tag{A.202}$$

where

$$n_{q_{ij}} = \begin{pmatrix} 0 & n_{p_i} & 0 & n_{p_j} \end{pmatrix} \tag{A.203}$$

$$n_{q'_{ij}} = \begin{pmatrix} 0 & n_{p_{i'}} & 0 & n_{p_{j'}} \end{pmatrix} \tag{A.204}$$

and, using the notation $\xi_k = A_k {\xi'}_k$, $k \in \{i, j\}$, $\xi \in \{f, g\}$,

$$\begin{aligned} n_{p_i} &= (k_\theta + N_\theta)\Big((f_i{}^\top f_j) f_j{}^\top B(p_i, g'_i) - (g_i{}^\top f_j) f_j{}^\top B(p_i, f'_i)\Big) \\ &+ (c_\theta + N_{\theta'})\Big({p_j}'^\top B^\top(p_j, f'_j)\big(f_i{}^\top f_j B(p_i, g'_i) - g_i{}^\top f_j B(p_i, f'_i)\big)\Big) \\ &+ f_j{}^\top (f_i{}^\top f_j) B(p_i', g'_i) - f_j{}^\top (g_i{}^\top f_j) B(p_i', f'_i) \end{aligned} \tag{A.205}$$

$$\begin{aligned} n_{p_j} &= (k_\theta + N_\theta)\Big((f_i{}^\top f_j) g_i{}^\top B(p_j, f'_j) - (g_i{}^\top f_j) f_i{}^\top B(p_j, f'_j)\Big) \\ &+ (c_\theta + N_{\theta'})\Big({p_i}'^\top \big(f_i{}^\top f_j B(p_i, g'_i) - g_i{}^\top f_j B(p_i, f'_i)\big)\Big)^\top B(p_j, f'_i) \\ &+ g_i{}^\top (f_i{}^\top f_j) B(p_i', g'_i) - f_i{}^\top (g_i{}^\top f_j) B(p_i', f'_i) \end{aligned} \tag{A.206}$$

$$n_{p_{i'}} = (c_\theta + N_{\theta'})\Big((f_i{}^\top f_j) f_j{}^\top B(p_i, g'_i) - (g_i{}^\top f_j) f_j{}^\top B(p_i, f'_i)\Big) \tag{A.207}$$

$$n_{p_{j'}} = (c_\theta + N_{\theta'})\Big((f_i{}^\top f_j) g_i{}^\top B(p_j, f'_j) - (g_i{}^\top f_j) f_i{}^\top B(p_j, f'_j)\Big) \tag{A.208}$$

$$
\begin{aligned}
(n_{q'_{ij}}\gamma)_{q_{ij}} &= (c_\theta + N_{\theta'})\Big(- \big({f'_j}^\top {A_j}^\top B_i \gamma^{i,p} \\
&+ {f'_i}^\top {A_i}^\top B_j \gamma^{j,p}\big) \begin{pmatrix} 0 & {f'_j}^\top {A_j}^\top B_{g_i} & 0 & {g'_i}^\top {A_i}^\top B_j \end{pmatrix} \\
&- ({g'_i}^\top {A_i}^\top A_j f'_j) \begin{pmatrix} 0 & \lambda_f & 0 & \lambda_\gamma \end{pmatrix} \\
&+ ({f'_j}^\top {A_j}^\top B_{g_i} \gamma^{i,p} \\
&+ {g'_i}^\top {A_i}^\top B_j \gamma^{j,p}) \begin{pmatrix} 0 & {f'_j}^\top {A_j}^\top B_i & 0 & {f'_i}^\top {A_i}^\top B_j \end{pmatrix} \\
&+ ({f'_i}^\top {A_i}^\top A_j f'_j) \begin{pmatrix} 0 & \lambda_g & 0 & \lambda_{\gamma g} \end{pmatrix} \Big)
\end{aligned} \tag{A.209}
$$

$$
\begin{aligned}
\lambda_f &= {f'_j}^\top {A_j}^\top B(\gamma^{i,p}, f'_i) + {\gamma^{j,p}}^\top {B_j}^\top B_i \\
\lambda_\gamma &= {\gamma^{i,p}}^\top {B_i}^\top B_j + {f'_i}^\top {A_i}^\top B(\gamma^{j,p}, f'_j) \\
\lambda_g &= {f'_j}^\top {A_j}^\top B(\gamma^{i,p}, g'_i) + {\gamma^{j,p}}^\top {B_j}^\top B_{g_i} \\
\lambda_{\gamma g} &= {\gamma^{i,p}}^\top {B_{g_i}}^\top B_j + {g'_i}^\top {A_i}^\top B(\gamma^{j,p}, f'_j)
\end{aligned}
$$

A.9 Derivatives with Respect to Force Related Parameters

In this section, partial derivatives of force terms with respect to force related model parameters are presented [50]. Derivatives of Q_i^A of Eq. (A.164) with respect to parameters F_i^A and ${s'}_i^C$ are

$$
\begin{pmatrix} F_i^A \\ 2{G_i}^\top {n'}_i^A \end{pmatrix}_{F_i^A} = \begin{pmatrix} I \\ 2{G_i}^\top {n'}_{i\,F_i^A}^A \end{pmatrix} \tag{A.210}
$$

$$
\begin{pmatrix} \hat{F}_i^A \\ 2\hat{G}_i^\top {n'}_i^A \end{pmatrix}_{{s'}_i^C} = \begin{pmatrix} \hat{F}_i^A \\ 2\hat{G}_i^\top {n'}_i^A \end{pmatrix}_{{s'}_i^C} = \begin{pmatrix} 0 \\ 2{G_i}^\top ({n'}_i^A)_{{s'}_i^C} \end{pmatrix} \tag{A.211}
$$

Note that F_i^A and ${n_i'}^A$ are related through Eq. (A.110); e.g., if F_i^A is constant, then ${n_i'}^A = \tilde{s'}_i^C {A_i}^\top F_i^A$ and ${n_i'}^A{}_{F_i^A} = \tilde{s'}_i^C {A_i}^\top$.

Derivatives of TSDA forces with respect to model parameters

$$\beta \in \{s_i'^P, s_j'^P, k, c, l_0, F\}$$

are [50]

$$Q_{i\,s_i'^P}^{TSDA} = \begin{pmatrix} \left(\frac{f}{l} d_{ij}\right)_{s_i'^P} \\ B^\top(p_i, s_i'^P) \left(\frac{f}{l} d_{ij}\right)_{s_i'^P} + \frac{f}{l} R_{k=1,\dots,4} \left({d_{ij}}^\top N(p_i, e_k)\right) \end{pmatrix} \tag{A.212}$$

$$Q_{i\,s_j'^P}^{TSDA} = \begin{pmatrix} \left(\frac{f}{l} d_{ij}\right)_{s_j'^P} \\ B^\top(p_i, s_i'^P) \left(\frac{f}{l} d_{ij}\right)_{s_j'^P} \end{pmatrix} \tag{A.213}$$

$$Q_{j\,s_i'^P}^{TSDA} = \begin{pmatrix} \left(-\frac{f}{l} d_{ij}\right)_{s_i'^P} \\ -B^\top(p_j, s_j'^P) \left(\frac{f}{l} d_{ij}\right)_{s_i'^P} \end{pmatrix} \tag{A.214}$$

$$Q_{j\,s_j'^P}^{TSDA} = \begin{pmatrix} \left(-\frac{f}{l} d_{ij}\right)_{s_j'^P} \\ -B^\top(p_j, s_j'^P) \left(\frac{f}{l} d_{ij}\right)_{s_j'^P} - \frac{f}{l} R_{k=1,\dots,4} \left({d_{ij}}^\top N(p_j, e_k)\right) \end{pmatrix} \tag{A.215}$$

and

$$Q_{i\beta}^{TSDA} = \begin{pmatrix} \frac{d_{ij}}{l} \left(k_\beta (l - l_0) - k l_{0\beta} + c_\beta l' + F_\beta\right) \\ B^\top(p_i, s_i'^P) \frac{d_{ij}}{l} \left(k_\beta (l - l_0) - k l_{0\beta} + c_\beta l' + F_\beta\right) \end{pmatrix} \tag{A.216}$$

$$Q_{j\beta}^{TSDA} = \begin{pmatrix} -\frac{d_{ij}}{l} \left(k_\beta (l - l_0) - k l_{0\beta} + c_\beta l' + F_\beta\right) \\ -B^\top(p_j, s_j'^P) \frac{d_{ij}}{l} \left(k_\beta (l - l_0) - k l_{0\beta} + c_\beta l' + F_\beta\right) \end{pmatrix} \tag{A.217}$$

for $\beta \in \{k, c, l_0, F\}$, where

$$\begin{aligned}\left(\frac{f}{l}d_{ij}\right)_{\beta} &= -\frac{f}{l^3}d_{ij}{d_{ij}}^{\top}d_{ij\beta} + \frac{d_{ij}}{l}\Big(\frac{k+F_l}{l}{d_{ij}}^{\top}d_{ij\beta} \\ &+ (c+F_{l'})\Big(-\frac{1}{l^3}{d_{ij}}^{\top}d_{ij}{}'{d_{ij}}^{\top}d_{ij\beta} + \frac{1}{l}{d_{ij}}^{\top}d_{ij\beta}' + \frac{1}{l}{d_{ij}}'^{\top}d_{ij\beta}\Big)\Big) \\ &+ \frac{f}{l}d_{ij\beta} + (k_\beta(l-l_0) - kl_{0\beta} + c_\beta l' + F_\beta)\frac{d_{ij}}{l} \end{aligned} \tag{A.218}$$

for $\beta \in \{s_i'^P, s_j'^P, k, c, l_0, F\}$, and

$$d_{ij}{}_{s_i'^P} = -A_i \tag{A.219}$$

$$d_{ij}{}_{s_j'^P} = A_j \tag{A.220}$$

$${d_{ij}}'{}_{s_i'^P} = -N(p_i, {p_i}') \tag{A.221}$$

$${d_{ij}}'{}_{s_j'^P} = N(p_j, {p_j}') \tag{A.222}$$

$$d_{ij}{}_{k,c,l_0,F} = 0 \tag{A.223}$$

$$k_{s_i'^P, s_j'^P, c, l_0, F} = 0 \tag{A.224}$$

$$c_{s_i'^P, s_j'^P, k, l_0, F} = 0 \tag{A.225}$$

$$l_{0\,s_i'^P, s_j'^P, k, c, F} = 0 \tag{A.226}$$

$$F_{s_i'^P, s_j'^P, k, c, l_0} = 0 \tag{A.227}$$

Derivatives of RSDA forces with respect to model parameters

$$\beta \in \{f_i', f_j', g_i', h_i', h_j', k, c, N\}$$

are [50]

$$Q_{i_{h_i'}}{}^{RSDA} = \begin{pmatrix} 0 \\ \\ 2nG_i{}^{\top} \end{pmatrix} \tag{A.228}$$

$$Q_{i_{h'_j}}{}^{RSDA} = 0 \tag{A.229}$$

$$Q_{j_{h'_i}}{}^{RSDA} = 0 \tag{A.230}$$

$$Q_{j_{h'_j}}{}^{RSDA} = \begin{pmatrix} 0 \\ -2nG_j{}^{\top} \end{pmatrix} \tag{A.231}$$

and

$$Q_{i_\beta}{}^{RSDA} = \begin{pmatrix} 0 \\ 2G_i{}^{\top}h'_i n_\beta \end{pmatrix} \tag{A.232}$$

$$Q_{j_\beta}{}^{RSDA} = \begin{pmatrix} 0 \\ -2G_j{}^{\top}h'_j n_\beta \end{pmatrix} \tag{A.233}$$

for $\beta \in \{f'_i, f'_j, g'_i, k_\theta, c_\theta, N\}$, where

$$n_\beta = (k_\theta + N_\theta)\theta_\beta + (c_\theta + N_{\theta'})\theta'_\beta + k_{\theta\beta}(\theta + 2n_{rev}\pi) + c_{\theta\beta}\theta' + N_\beta \tag{A.234}$$

for $\beta \in \{f'_i, f'_j, g'_i, k_\theta, c_\theta, N\}$, and

$$\theta_{f'_i} = -(g'_i{}^{\top}A_i{}^{\top}A_j f'_j)f'_j{}^{\top}A_j{}^{\top}A_i \tag{A.235}$$

$$\begin{aligned} \theta'_{f'_i} &= (f'_j{}^{\top}A_j{}^{\top}B(p_i, g'_i)p_i{}' + g'_i{}^{\top}A_i{}^{\top}B(p_j, f'_j)p_j{}')f'_j{}^{\top}A_j{}^{\top}A_i \\ &- (f'_j{}^{\top}A_j{}^{\top}N(p_i, p_i{}') + p_j{}'^{\top}B^{\top}(p_j, f'_j)A_i)g'_i{}^{\top}A_i{}^{\top}A_j f'_j \end{aligned} \tag{A.236}$$

$$\theta_{f'_j} = -(g'_i{}^{\top}A_i{}^{\top}A_j f'_j)f'_i{}^{\top}A_i{}^{\top}A_j + (f'_i{}^{\top}A_i{}^{\top}A_j f'_j)g'_i{}^{\top}A_i{}^{\top}A_j \tag{A.237}$$

$$\begin{aligned} \theta'_{f'_j} &= \big((f'_i{}^{\top}A_i{}^{\top}A_j f'_j)p_i{}'^{\top}B^{\top}(p_i, g'_i) + f'_j{}^{\top}A_j{}^{\top}B(p_i, g'_i)p_i{}'f'_i{}^{\top}A_i{}^{\top} \\ &- f'_j{}^{\top}A_j{}^{\top}B(p_i, f'_i)p_i{}'g'_i{}^{\top}A_i{}^{\top} - (g'_i{}^{\top}A_i{}^{\top}A_j f'_j)p_i{}'^{\top}B^{\top}(p_i, f'_i) \\ &+ g'_i{}^{\top}A_i{}^{\top}B(p_j, f'_j)p_j{}'f'_i A_i{}^{\top} - f'_i{}^{\top}A_i{}^{\top}B(p_j, f'_j)p_j{}'g'_i{}^{\top}A_i{}^{\top}\big)A_j \\ &+ \big((f'_i{}^{\top}A_i{}^{\top}A_j f'_j)g'_i{}^{\top}A_i{}^{\top} - (g'_i{}^{\top}A_i{}^{\top}A_j f'_j)f'_i{}^{\top}A_i{}^{\top}\big)N(p_j, p_j{}') \end{aligned} \tag{A.238}$$

$$\theta_{g_i'} = ({f_i'}^\top {A_i}^\top A_j f_j') {f_j'}^\top {A_j}^\top A_i \tag{A.239}$$

$$\begin{aligned} \theta'_{g_i'} &= ({f_i'}^\top {A_i}^\top A_j f_j') {f_j'}^\top {A_j}^\top N(p_i, {p_i}') - {f_j'}^\top {A_j}^\top B(p_i, f_i') {p_i}' {f_j'}^\top {A_j}^\top A_i \\ &+ {p_j}'^\top B^\top(p_j, f_j') \big(({f_i'}^\top {A_i}^\top A_j f_j') - A_i f_i' {f_j'}^\top {A_j}^\top \big) A_i \end{aligned} \tag{A.240}$$

$$\theta_{k_\theta, c_\theta, N} = 0 \tag{A.241}$$

$$\theta'_{k_\theta, c_\theta, N} = 0 \tag{A.242}$$

$$k_{\theta\, f_i', f_j', g_i', c_\theta, N} = 0 \tag{A.243}$$

$$c_{\theta\, f_i', f_j', g_i', k_\theta, N} = 0 \tag{A.244}$$

$$N_{f_i', f_j', g_i', k_\theta, c_\theta} = 0 \tag{A.245}$$

REFERENCES

[1] C. Amza, A.L. Cox, S. Dwarkadas, L.J. Jin, K. Rajamani, and W. Zwaenepoel. Adaptive Protocols for Software Distributed Shared Memory. *Proceedings of the IEEE, Special Issue on Distributed Shared Memory*, 1999.

[2] M. Anantharaman and M. Hiller. Numerical Simulation of Mechanical Systems Using Methods for Differential Algebraic Equations. *International Journal for Numerical Methods in Engineering*, 32:1531–1542, 1991.

[3] L. Ascher, U.and Petzold. Projected collocation for higher-order higher-index differential-algebraic equations. *JCAM J. Sci. Comput.*, 1992.

[4] U. Ascher, R. Mattheij, and R. Russell. *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*. SIAM, 1995.

[5] U. Ascher and L. Petzold. Stability of Computational Methods for Constrained Dynamics Systems. *SIAM J. Sci. Comput.*, 14:95–120, 1993.

[6] U. Ascher and L. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. SIAM, 1998.

[7] U. Ascher and R. Spiteri. Collocation Software for Boundary Value Differential-Algebraic Equations. *SIAM J. Sci. Comput.*, 15:938–952, 1995.

[8] K. Atkinson. *Introduction to Numerical Analysis*. Wiley, New York, second edition, 1989.

[9] K. Atkinson and W. Han. *Theoretical Numerical Analysis. A Functional Analysis Framework*. Springer-Verlag, 2001.

[10] O.L.1 Bandman. Fine-Grained Parallelism in Computational Mathematics. 27(4):170–182, 2001.

[11] E. Bayo, J. Cardenal, J. Cuadrado, and P. Morer. Intelligent Simulation of Multibody Dynamics: Space-State and Descriptor Methods in Sequential and Parallel Computing Environments. *Multibody System Dynamics*, 4:55–73, 2000.

[12] G.E. Blelloch, P.B. Gibbons, and Y. Matias. Provably efficient scheduling for languages with fine-grained parallelism. *Journal of the ACM (JACM)*, 46(2):144–155, 1999.

[13] K.E. Brenan, S.L. Campbell, and L.R. Petzold. *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*. SIAM, second edition, 1996.

[14] K. Burrage. *Parallel and Sequential Methods for Ordinary Differential Equations*. Clarendon Press - Oxford, 1995.

[15] D. R. Butenhof. *Programming with POSIX Threads*. Addison-Wesley, 1997.

[16] Y. Cao, S. Li, L. Petzold, and R. Serban. Adjoint Sensitivity Analysis for Differential-Algebraic Equations: The Adjoint DAE System and its Numerical Solution. *SIAM J. Sci. Comput*, 24(3):1076–1089, 2003.

[17] M. Caracotsios and W.E. Stewart. Sensitivity analysis of initial value problems with mixed ODEs and algebraic constraints. *Comput. Chem. Engrg.*, 9:359–365, 1985.

[18] Chi-Tsong Chen. *Linear Systems, Oxford Series in Electrical and Computer Engineering*. Oxford University Press, third edition, 1999.

[19] S. Dwarkadas, P. Keleher, A. Cox, and W Zwaenepcd. Evaluation of Release Consistent Software Distributed Shared Memory on Emerging Network Technology. *Proceedings of the 20th Annual International Symposium on Computer Architecture*, pages 144–155, 1993.

[20] W.H. Enright and D.J. Higham. Parallel defect control. *BIT*, 31:647–663, 1991.

[21] W.F. Feehery, J.E. Tolsma, and P.I. Barton. Efficient sensitivity analysis of large-scale differential-algebraic systems. *Applied Numerical Mathematics*, 25, 1997.

[22] C. Fuhrer and B.J. Leimkuhler. Numerical solution of differential-algebraic equations for constrained mechanical motion. *Numer. Math*, 59:55–69, 1991.

[23] F.R. Gantmacher. *Matrix Theory*, volume 1. Chelsea Pub Co., second edition, 1990.

[24] C.W. Gear. Differential-Algebraic Equation Index Transformations. *SIAM J. Sci. Stat. Comput.*, 9:39–47, 1988.

[25] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II. Stiff and Differential-Algebraic Problems*. Springer-Verlag, second revised edition edition, 1996.

[26] E.J. Haug. Design Sensitivity Analysis of Dynamic Systems. In C.A. Mota-Soares, editor, *Computer-Aided Design: Structural and Mechanical Systems*, Berlin, 1987. Springer-Verlag.

[27] E.J. Haug. *Computer-Aided Kinematics and Dynamics of Mechanical Systems, Volume I: Basic Methods*. Allyn and Bacon, Needham Heights, Massachusetts, 1989.

[28] E.J. Haug, S.C. Wu, and S.M. Yang. Dynamics of Mechanical Systems with Coulomb Friction, Stiction, Impact and Constraint Addition-Deletion-I. *Mechanism and Machine Theory*, 21(5):401–406, 1986.

[29] J.L. Hennessy and D.A. Patterson. *Computer Architecture A Quantitative Approach*. Morgan Kaufmann Publishers, Inc., second edition edition, 1986.

[30] A. C. Hindmarsh, P. N. Brown, K. E. Grant, S. L. Lee, R. Serban, D. E. Shumaker, and C. S. Woodward. SUNDIALS: Suite of Nonlinear and Differential/Algebraic Equation Solvers. *ACM Transactions on Mathematical Software*, 2003.

[31] A. C. Hindmarsh and A. G. Taylor. User Documentation for IDA, A Differential-Algebraic Equation Solver for Sequential and Parallel Computers. 1999.

[32] L.O. Jay. Inexact Simplified Newton Iterations for Implicit Runge-Kutta Methods. 1998.

[33] L.O. Jay. Structure preservation for constrained dynamics with Super Partitioned Additive Runge-Kutta Methods. *SIAM J. Sci. Comput.*, 20(2):416–446, 1998.

[34] L.O. Jay and T. Braconnier. A Parallelizable Preconditioner For the Iterative Solution of Implicit Runge-Kutta Type Methods. 1999.

[35] Ch Lubich. On projected Runge-Kutta methods for differential-algebraic equations. *BIT*, 31:545–550, 1991.

[36] A. Lumsdaine and M.W. Reichelt. Waveform Iterative Techniques for Device Transient Simulation on Parallel Machines. *Proc. Sixth SIAM Conference on Parallel Processing for Scientific Computing*, 237-245, 1993.

[37] T. Maly and L. R. Petzold. Numerical methods and software for sensitivity analysis of DAE. *Applied Numerical Mathematics*, 20, 1996.

[38] E. P. Markatos and T. J. LeBlanc. Using processor affinity in loop scheduling on

shared-memory multiprocessors. *Proceedings of the 1992 ACM/IEEE conference on Supercomputing*, 1992.

[39] R. Marz. Practical Lyapunov Stability Criteria for Differential Algebraic Equations. *Numerical Analysis and Mathematical Modelling Banach Center Publications*, 29, 1994.

[40] T. Muir. *A Treatise on the Theory of Determinants*. Dover Publications, Inc., New York, 1960.

[41] D. Negrut. *On the Implicit Integration of Differential-Algebraic Equations of Multibody Dynamics*. PhD thesis, University of Iowa, 1998.

[42] D. Negrut and E.J. Haug. State-space based implicit integration of the differential-algebraic equations of multibody dynamics. *Proceedings of the 1999 ASME Design Engineering Technical Conference, September 12-15, Las Vegas, Nevada*, 1999.

[43] P.E. Nikravesh. *Computer-Aided Analysis of Mechanical Systems*. Prentice Hall, Englewood Cliffs, NJ 07632, 1988.

[44] V. Olariu. *Analiza Matematica*. Editura Didactica si Pedagogica, Bucuresti, 1981.

[45] P.S. Pacheco. *Parallel Programming with MPI*. Morgan Kaufmann, 1996.

[46] L.R. Petzold. Differential-Algebraic Equations Are Not ODE's. *SIAM Journal of Scientific and Statistical Computing*, 3(3):367–384, 1982.

[47] F.A. Potra and W.C. Rheinboldt. Differential-geometric techniques for solving differential algebraic equations. *Real-Time Integration of Mechanical System Simulation*, 155-191, 1990.

[48] P.J. Rabier and W.C. Rheinboldt. Nonholonomic Motion of Rigid Mechanical Systems from a DAE viewpoint. *SIAM*, 2000.

[49] P.J. Roosta. Parallel Processing and Parallel Algorithms: Theory and Computation. *Springer-Verlag*, 2000.

[50] R. Serban. *Dynamic and Sensitivity Analysis of Multibody Systems*. PhD thesis, University of Iowa, 1998.

[51] R. Serban and E.J. Haug. Analytical Derivatives for Multibody System Analyses. *Mechanics of Structures and Machines*, 26(2):145–173, 1998.

[52] R. Serban, D. Negrut, E.J. Haug, and F.A. Potra. A Topology Based Approach for Exploiting Sparsity in Multibody Dynamics in Cartesian Formulation. *Mechanics of Structures and Machines*, 25(3):379–396, 1997.

[53] Y. Stver. Collocation methods for solving linear differential algebraic boundary value problems. 1991.

[54] A.S. Tanenbaum. *Modern Operating Systems*. Prentice-Hall, Inc., second edition edition, 2001.

[55] P.J. van der Houwen and W.A. van der Veen. *Solving implicit differential equations on parallel computers, Technical Report*. 1995.

[56] P.J. van der Houwen and W.A. van der Veen. *Waveform relaxation methods for implicit differential equations, Technical Report*. 1996.

[57] D. Vanderstraeten. A Stable and Efficient Parallel Block Gram-Schmidt Algorithm. In P. Amestoy et al., editor, *Euro-Par'99, LNCS 1685*, Berlin, 1999. Springer-Verlag.

[58] R.A. Wehage and Haug E.J. Generalized Coordinate Partitioning for Dimension Reduction in Analysis of Constrained Dynamic Systems. *ASME Journal of Mechanical Design*, 104:247, 1982.

[59] J. White and Vincentelli A.S. Waveform Relaxation: Theory and Practice. *Transactions of The Society for Computer Simulation*, 2(1):95–133, 1985.