Multibody System Dynamics (2024) 64:1–31
https://doi.org/10.1007/s11044-024-10030-4

RESEARCH

Simultaneous optimal system and controller design for multibody systems with joint friction using direct sensitivities

Adwait Verulkar[1] · Corina Sandu[1] · Adrian Sandu[2] · Daniel Dopico[3]

Received: 25 December 2023 / Accepted: 26 September 2024 / Published online: 22 October 2024

Abstract

Real-world multibody systems are often subject to phenomena like friction, joint clearances, and external events. These phenomena can significantly impact the optimal design of the system and its controller. This work addresses the gradient-based optimization methodology for multibody dynamic systems with joint friction using a direct sensitivity approach. The Brown–McPhee model has been used to characterize the joint friction in the system. This model is suitable for the study due to its accuracy for dynamic simulation and its compatibility with sensitivity analysis. This novel methodology supports codesign of the multibody system and its controller, which is especially relevant for applications like robotics and servo-mechanical systems, where the actuation and design are highly dependent on each other. Numerical results are obtained using a software package written in Julia with state-of-the-art libraries for automatic differentiation and differential equations. Three case studies are provided to demonstrate the attractive properties of simultaneous optimal design and control approach for certain applications.

Keywords Sensitivity analysis · Optimal design · Optimal control · Julia · Differential equations · Computational efficiency · Automatic differentiation

✉ A. Verulkar
adwaitverulkar@vt.edu

C. Sandu
csandu@vt.edu

A. Sandu
asandu7@vt.edu

D. Dopico
ddopico@udc.es

[1] Terramechanics, Multibody and Vehicle Systems Laboratory, Department of Mechanical Engineering, Virginia Tech, Blacksburg, USA

[2] Computational Science Laboratory, Department of Computer Science, Virginia Tech, Blacksburg, USA

[3] Laboratorio de Ingeniería Mecánica, Department of Naval and Industrial Engineering, University of A Coruña, A Coruña, Spain

1 Introduction

Performance analysis of mechanical systems is usually conducted by studying the dynamics through a simulation software. Due to advances in differential equation solvers and optimization techniques, it has become increasingly convenient to perform sensitivity analysis and gradient-based dynamic optimization on large-scale dynamical systems involving multiple design/control parameters. Multibody systems are special types of dynamic systems that consist of multiple links connected by joints. This gives rise to relatively complex systems of equations that require specific numerical solvers. Recent research on multibody systems has been focused on simulating realistic behaviors in these systems such as friction [39, 44, 54, 64], clearances [27, 28, 62, 78], events [20, 21], and even a combination of these [29]. Such behaviors further add to the complexity of the resulting system of equations.

The two predominant approaches for sensitivity analysis for dynamic systems are the direct/forward and adjoint/reverse approaches. The direct sensitivity approach [16, 23, 79] is ideal when the gradients of several outputs need to be computed with respect to a relatively small number of parameters. This method involves *directly* differentiating the dynamic model with respect to the system parameters, which yields a tangent linear model (TLM). The TLM can then be integrated to obtain the *direct* sensitivities of the model, which are the derivatives of the original system states with respect to its parameters. The gradient of any output functional, such as an optimization objective, can then be obtained using these sensitivities by applying the chain rule of differentiation. In contrast, the adjoint/reverse sensitivity approach is more advantageous when the number of parameters is large compared to the number of output functionals. Rather than explicitly solving for system sensitivities, this method constructs an additional system using the Lagrangian of the original optimization problem. This additional system is *adjoint* to the original one and is solved backwards in time. The solution of this adjoint system can then be used to compute the gradient of the output functional with respect to any set of parameters of the system.

Research on the sensitivity analysis of multibody systems has evolved into several approaches over the past few decades. A detailed literature review of multibody and corresponding sensitivity formulations developed over the years can be found in [80]. Haug [37] developed the sensitivity methodology for multibody systems using Lagrangian index-3 and index-1 formulations. Chang and Nikravesh [18] experimented with Baumgarte stabilization [7] to prevent constraint violation. Aforementioned approaches use the differential-algebraic equation (DAE) form of the multibody formulations. An approximate ordinary differential equation (ODE) formulation for multibody systems is the penalty approach that minimizes the action integral of the system. Sensitivity methodologies using the penalty approach have been developed for both direct and adjoint sensitivity analysis [63, 85]. Dopico et al. [23] presented a sensitivity analysis approach using the exact ODE representation of a multibody system obtained through Maggi's equations. Additionally, event-based sensitivity formulations for mulitbody systems have been presented by Corner et al. [20, 21]. Valera et al. [51] described a discrete method for adjoint sensitivity analysis using the augmented Lagrangian index-3 formulation with projections (ALI3-P). Among other applications, sensitivity analysis can also be employed for parameter estimation. Blanchard et al. [10–12] presented his work on parameter estimation for mechanical systems with uncertain parameters using polynomial chaos representation of uncertainty.

Sensitivity analysis and, consequently, gradient-based optimization are widely employed due to their notable efficiency and speed compared to gradient-free search methods like the Nelder–Mead method [60], as well as evolutionary optimization techniques such as the genetic algorithm [43] and particle swarm optimization [46]. However, the application of direct sensitivity analysis to multibody systems with friction requires several considerations to be made in terms of solution techniques, accuracy and stability of the sensitivities, and computational efficiency. Optimization is inherently iterative in nature, requiring the system dynamics model and TLMs to be repeatedly solved. This process is especially challenging when dealing with multibody systems with friction, due to the implicit and stiff nature of the dynamic and sensitivity equations involved. Hence it becomes imperative to develop a generic and computationally efficient optimization methodology to skillfully tackle these complexities.

Many dynamic systems rely on active control to achieve specific trajectory goals. In the realm of nonlinear optimal control, it is a common practice to parameterize the control function. One popular approach involves using basis functions along with their corresponding coefficients as control parameters [68, 82]. In contrast, state/output feedback control typically involves a fixed number of control parameters. Frequently, systems must be customized for particular applications, necessitating optimization in both design and control aspects. Recent research has underscored the advantages of simultaneous optimization over separate design and control optimization. This approach, also known as codesign, has the potential to yield more efficient systems by exploiting the flexibility in design to find efficient control solutions, compared to fixed design approaches. When the optimization methodology treats design and control parameters on equal footing, it can harness this potential afforded by codesign. The following references highlight the significance of simultaneous design and control in specific applications.

- In certain aerospace applications where there are extreme power-to-weight concerns, it might be beneficial to do simultaneous optimization of control and design [59]. In this work, codesign is shown to be helpful for the integration of energy management optimization along with optimal vehicle sizing for a hybrid electric propulsion aircraft.
- In civil engineering applications, building structures for earthquake resistance can be done through simultaneous design and control [3]. This type of structure uses active controls to prevent structural damage due to seismic activity.
- Specifically talking about multibody systems with friction, codesign optimization strategies are also used for design of active suspension systems [4]. This method is exactly the same as using basis functions to convert a continuous control into discrete parameterizations, referred to as direct transcription.

Simultaneous design and control are especially useful in legged robotic systems. Earlier robots employed high-gain feedback and therefore used considerable joint torque to cancel out the natural dynamics of the machine to follow a desired trajectory. Optimizing the control without making any design considerations may lead to suboptimal design choices. An example of such a system is ASIMO, which is a legged-robot that uses roughly 20 times the energy (scaled) that a human uses to walk on a flat surface, as measured by the cost of transport [19]. Leveraging the design and dynamics to suit the application's control needs is crucial for such systems [26]. This paper deals with multibody formulations that are free of events such as impacts, collisions, and intermittent joint contacts. These restrictions enable the use of smooth formulations. However, legged-robotic systems may benefit from the use of a contact-based friction models [22]. Most control algorithms on robotic systems use optimization in the form of model predictive control. Running optimization on a contact-based friction model is challenging due to the fully nonsmooth formulation. An alternative approach is to consider a piecewise continuous trajectory for the gait motion and applying energy and momentum conservation when changing step.

This paper presents the methodology for direct sensitivity analysis and dynamic optimization of multibody systems with joint friction. By converting the continuous control signal into a parameterized form, the methodology can be applied to codesign case studies as well. The direct sensitivity approach for gradient computation is efficient and outperforms reverse-mode (adjoint) approach for optimization problems involving a *relatively small* number of design/control parameters in comparison to the number of state variables. The formulation uses centroidal body-fixed reference frames with the orientation of the bodies defined using the Euler parameters. The resulting matrices involved in the equations of motion are sparse, thereby allowing for efficient solution techniques such as Newton–Krylov method [45], generalized minimal residual method (GMRES) [6], and Jacobian-free Newton–Krylov method [48]. Three case studies are provided that apply the methodology for a pure control optimization example of inverted pendulum, a pure design optimization example of a governor mechanism, and a codesign example of a spatial slider-crank mechanism. The novel contributions of this paper are summarized as follows.

1. The work develops a direct sensitivity-based optimization approach for multibody dynamic systems with Brown–McPhee joint friction.
2. A codesign methodology for simultaneous optimal design of system and controller parameters is introduced. The optimal design problem is constrained by the system dynamics with joint friction.
3. The effectiveness of methodology is demonstrated through real-world case studies and validated through numerical results.
4. The MBSVT 2.0 software package has been developed in Julia for sensitivity analysis and optimization of multibody systems. The implementation leverages recent advances in differentiable programming and modern interfaces to differential equation solvers.

The remainder of the paper is organized as follows. Section 2 discusses the computational aspects of the work. Section 3 reviews the development of equations of motion for multibody systems with friction. Section 4 reviews bound-constrained optimization using direct sensitivity analysis. Section 5 provides the numerical validation of the methodology using various case studies. Finally, Sect. 6 draws conclusions and points to future work.

2 Computational aspects of multibody simulations and differentiation

Recent advances in automatic differentiation (AD) have started a discussion on discrete versus continuous sensitivity analysis methods [15, 52, 73]. Since the 1990s, AD has been a popular tool for sensitivity computation in ODEs/DAEs and, by extension, for multibody systems as well [9, 25, 50, 67]. Conventionally, it has been a practice to differentiate the dynamic equations of motion to obtain the sensitivity equations. As stated in Sect. 1, the sensitivity equations are obtained through direct differentiation of system dynamics. These sensitivity equations, also known as TLMs, turn out to be differential equations that can be integrated to obtain the model sensitivities with respect to the design or control parameters. However, in recent times, another course of action is to simply perform AD of the differential equation solver step itself [69]. The discrete sensitivity approach is mathematically equivalent to the previous continuous approach. However, in certain cases, it may offer improved ease of programming without sacrificing computational efficiency, the main reason being the ability of AD to perform compiler optimizations by making use of structure between the primal and derivative constructions. This allows the AD code to perform single instruction on multiple data (SIMD), constant folding, and common subexpression elimination (CSE) [58]. An experienced user can program these optimizations manually. However, there lies a trade-off between the time and effort required to obtain an efficient code versus the computational cost saved. Additionally, there are differences in stability of continuous versus discrete approaches for reverse-mode AD, as noted in [47, 71]. It is important to note, however, that AD might not be a viable option in cases where the dynamic equations have to be implicitly solved. The iterative convergence required in implicit differential equation solvers creates unnecessary computations, which can be avoided in continuous approaches by leveraging the implicit function theorem. Also, AD of implicit solvers can only yield approximate derivative since the convergence is solved to a certain tolerance. A good application of AD is in building the derivative components of continuous sensitivity methods. These components can be obtained without computing a full Jacobian through efficient Jacobian-vector products. For multibody systems, most Jacobians and Jacobian-vector products required in dynamics and sensitivity computations are available as closed-form expressions, which are simplified to a large extent using domain-specific mathematical identities. This makes the derivative computation efficient to the point that it can be represented by a single nonallocating function call. Expressions involving Jacobians and Jacobian-vector products with respect to states can therefore be more efficiently computed through manual differentiation (MD), whereas those involving design or control parameters require AD. This not only alleviates the user's responsibility for providing Jacobian or Jacobian-vector products for specific multibody systems, but also facilitates space and time complexity. This hybrid fusion of MD for known derivatives and AD for unknown derivatives yields the benefits of both approaches.

Symbolic differentiation (SD) is another popular approach, which often exhibits similar time and space complexity to that of MD. However, most computer algebra systems do not have the capability to carry out expression simplifications in a sophisticated way that will take advantage of domain specific mathematical identities. This frequently results in complicated expressions, often referred to as expression swell, for higher-order derivatives and matrix calculus. A symbolic code frequently contains several auxiliary coefficients, which add to the computational cost. Also, some expressions and their derivatives cannot be generated through symbolic computations, like those involving large matrix inversions due to the prohibitively slow and memory intensive nature of symbolic computations. Symbolic differentiation also fails in cases where a function may not be represented by a mathematical formula and therefore cannot handle complex control flow further limiting its expressivity [8, 34]. Finally, there is the question of automated translation of a symbolic expression to an efficient numerical function. The user seldom has complete control over this process and may lead to suboptimal code. AD can handle all of these edge cases while the user being in greater control of the code.

Table 1 highlights the differences in computational cost using various differentiation approaches for some common terms required in multibody formulations such as the constraint Jacobian $\mathbf{\Phi_q}$ and various Jacobian-vector products like the total time derivative of the constraint vector $\mathbf{\Phi_q \dot{q}}$, acceleration term $(\mathbf{\Phi_q \dot{q}})_\mathbf{q} \mathbf{\dot{q}}$, and the generalized reaction term $\mathbf{\Phi_q^T \lambda}$. The matrices are computed for the slider-crank case study discussed in Sect. 5.3, which has a total of 20 constraints and 21 generalized coordinates. These matrices are required for each time step during the forward dynamics computation as well as the integration of the TLMs to obtain the direct sensitivities. Consequently, this operation needs to be as computationally efficient as possible to avoid long solution times.

As we can observe, MD performs best in terms of both space and time complexity when compared to AD and SD approaches. The speed comes from the fact that analytical Jacobian is computed effectively by a single function call (for sub-Jacobian computation) as opposed to the recursive function calls in the case of AD. However, AD can be made substantially

Table 1 Cost of different differentiation approaches for computing Jacobians of the slider-crank system discussed in Sect. 5.3

Operation	Manual	Automatic	Symbolic
$\mathbf{\Phi_q}$	21.992 μs / 23.59 kB	177.768 μs / 829.34 kB	1.042 ms / 101.12 kB
$\mathbf{\Phi_q \dot{q}}$	16.560 μs / 6.15 kB	152.169 μs / 224.50 kB	858.22 μs / 32.26 kB
$\mathbf{\Phi_q^T \lambda}$	30.254 μs / 15.22 kB	178.494 μs / 122.28 kB	995.59 μs / 38.26 kB
$\mathbf{(\Phi_q \dot{q})_q \dot{q}}$	18.223 μs / 20.79 kB	159.108 μs / 270.58 kB	938.31 μs / 40.66 kB

more efficient by employing sparsity detection and matrix coloring techniques [2, 32, 55] not employed in this analysis.

The results are obtained using a software package written in Julia for kinematic and dynamic simulation of multibody systems, direct sensitivity analysis, and gradient-based optimization. The software package takes advantage of recent advances in open-source Julia libraries such as differential equations solvers (`DifferentialEquations.jl`), optimization packages (`Optim.jl`), and support for forward and reverse mode AD (`ForwardDiff.jl`, `Zygote.jl`). Important functions for sensitivity analysis such as Jacobian matrices with respect to design variables, Jacobian-vector products, and gradients of objective functions are computed automatically and need not be user-provided. All derivative terms and sensitivity computations have been thoroughly validated using complex finite differences [81]. Moreover, due to native AD capability, the derivatives with respect to design parameters can be computed by the software. Despite being a relatively new programming language, Julia is rapidly gaining popularity within the scientific community for several compelling reasons. Its syntactical convenience, robust computational speed, and the myriad of packages dedicated to scientific computing and machine learning make it an attractive choice for researchers and practitioners alike. This provides flexibility to its users since multibody packages such as MSC Adams, SIMPACK, LMS VirtualLab Motion, RecurDyn, and Simscape are not open-source packages. Moreover, most of these packages focus on kinematics and dynamics capabilities, with sensitivity analysis and optimization taking the backstage. JModelica/Optimica (now known as Modelon OCT) [1], JuliaSim [70], CasADi/IPOPT [5, 83] are among the few tools that make substantial strides in this regard. These packages employ computational graph-based AD to extract gradient information for sensitivity analysis and optimization of dynamic systems. They are excellent packages capable of optimizing multibody systems but require some development effort on the user's end. Moreover, Julia's ease-of-development features, such as a built-in package manager, benchmarking tools, well-documented libraries, active user forums, cross-platform compatibility and compilation support, significantly facilitate contributions from new users. Other Julia libraries that can be employed for multibody dynamics and optimization are the acausal modeling package `Modia.jl` [24] and the package `RigidBodyDynamics.jl` [49]. These tools were not explored by the authors in this work but have been provided as alternatives that could potentially be used for optimization of multibody systems. `ModelingToolkit.jl`[53] is another acausal system model package that can be considered for this study. However, it currently lacks support for the type of differential-algebraic formulations used in this work. It also relies on `Symbolics.jl` [33] for differentiation and sparsity detection, necessitating symbolic traversability of all functions used to model the system.

3 Dynamics of multibody systems with friction

This section briefly covers the derivation of the equations of motion for multibody systems with joint friction using the index-1 DAE formulation. As it will become apparent by the end of this section, the friction forces ultimately depend on the state variables of the equation of motion due to its dependency on the normal force in the joint. This makes the dynamic equations of motion implicit and require special integration schemes to solve.

3.1 Joint friction forces

Computation of generalized friction force for multibody systems is a two-step process viz. computation of the magnitude of frictional force and torque at the joint, and, assembly of the generalized friction force vector. Friction can be modeled using several approaches. The approaches can be segregated based on whether they use a static or a dynamic model. This study uses the Brown–McPhee friction model [13] to describe joint friction. It is a quasi-static model governed by a single equation. Assuming that most mechanical systems have some lubrication, the friction between surfaces deviates from the dry Coulomb friction model. The mathematical representation of this friction model (excluding viscous friction) is as follows:

$$F_f(v,\boldsymbol{\mu}) = F_n\left[\mu_d \tanh\left(\frac{4v}{v_t}\right) + \frac{(\mu_s-\mu_d)\left(\frac{v}{v_t}\right)}{\left[\left(\frac{v}{2v_t}\right)^2+\frac{3}{4}\right]^2}\right]. \tag{1}$$

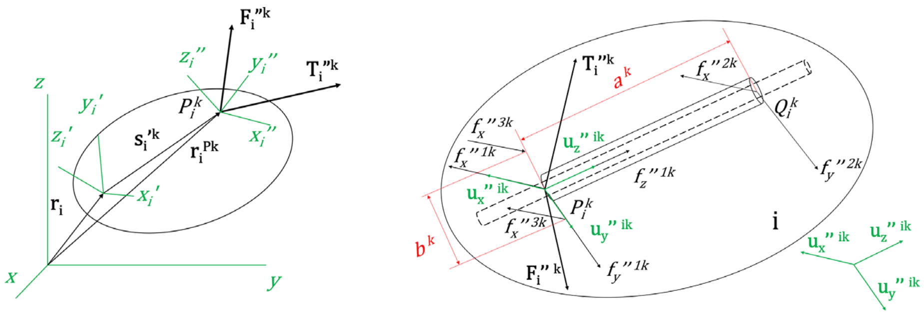
The main advantages of this model for sensitivity analysis of multibody systems are its C^1 continuity and differentiability and ability to simulate stiction by allowing relative motion at speeds lower than some user-defined threshold v_t. It is important to note that differentiating a friction model may not necessarily yield the same sensitivities as those obtained through piecewise continuous friction models like Coulomb. Haug [39] used the Brown–McPhee model for describing the joint friction between two bodies. Determining the transition velocity v_t poses a challenge, and according to Haug [39], it is advisable to set v_t to approximately ten times the average integration time step employed by the solver. A value of 10^{-2} to 10^{-3} was used in this work.

Another approach in modeling friction is using dynamic friction models. The Gonthier et al. friction model [31] is a more sophisticated model based on the LuGre [17] friction model. Unlike the Brown McPhee model, the Gonthier model incorporates dynamic states that require integration to compute the current friction. A detailed discussion of other friction models that are applicable for sensitivity analysis and optimization has been provided in [81].

3.2 Normal contact forces in joints

For a multibody system, the constraints are maintained during the motion of the system by internal reaction forces and torques as seen in Fig. 1a. The following equation can be used to calculate these physical quantities in the joint reference frame [39]:

$$\begin{Bmatrix} \mathbf{F}_i''^k \\ \mathbf{T}_i''^k \end{Bmatrix} = \begin{Bmatrix} \mathbf{C}_i^{k\mathrm{T}}\,\mathbf{A}_i^{\mathrm{T}}\,\boldsymbol{\Phi}_{\mathbf{r}_i}^{k\mathrm{T}}\boldsymbol{\lambda}^k \\ \mathbf{C}_i^{k\mathrm{T}}\left(\frac{1}{2}\,\mathbf{G}(\mathbf{p}_i)\,\boldsymbol{\Phi}_{\mathbf{p}_i}^{k\mathrm{T}} - \tilde{\mathbf{s}}_i'^k\,\mathbf{A}_i^{\mathrm{T}}\boldsymbol{\Phi}_{r_i}^{k\mathrm{T}}\right)\boldsymbol{\lambda}^k \end{Bmatrix}. \tag{2}$$



(a) Internal joint forces (b) Forces in a cylindrical joint

Fig. 1 Resolution of internal forces in joint reference frame

In equation (2), for a given body i, a holonomic joint k can be defined with the constraints $\mathbf{\Phi}^{\mathrm{k}} = \mathbf{0}$, $\mathbf{F}_i''^{k}$ and $\mathbf{T}_i''^{k}$ are the reaction force and moments in the joint coordinate frame, respectively, $\mathbf{C}_i^k$ and $\mathbf{A}_i$ are the joint-to-body and body-to-ground coordinate transformation matrices, respectively, $\mathbf{\Phi}^{\mathrm{k}}_{\mathbf{r}_i}$ and $\mathbf{\Phi}^{\mathrm{k}}_{\mathbf{p}_i}$ are the constraint Jacobians with respect to the Cartesian coordinates and Euler parameters of the ith body, respectively, $\boldsymbol{\lambda}^{\mathrm{k}}$ are the Lagrange multipliers associated with the constraints $\mathbf{\Phi}^{\mathrm{k}}$, and $\mathbf{s}_i'^{k}$ is the position vector in the ith body-fixed reference frame for the joint location. To calculate the effective joint normal force F_n, it is crucial to decompose the forces from the joint reference axes into their components. The joint torque generates a couple within the joint geometry, restricting rotational degrees of freedom and thereby influencing the effective joint normal force.

Haug [39] has presented the decomposition of forces and torques in cylindrical, translational, and revolute joints, as depicted in Fig. 1b. The clearances in the joint required for smooth motion are assumed to be negligible relative to the magnitude of link displacements. The force components are shown in Fig. 1b and can be expressed mathematically by the following equations:

$$f_x''^{1k} = -\mathbf{u}_x''^{ikT}\mathbf{F}_i''^{k} + \left(\frac{1}{a^k}\right)\mathbf{u}_y''^{ikT}\mathbf{T}_i''^{k}, \tag{3a}$$

$$f_y''^{1k} = -\mathbf{u}_y''^{ikT}\mathbf{F}_i''^{k} - \left(\frac{1}{a^k}\right)\mathbf{u}_x''^{ikT}\mathbf{T}_i''^{k}, \tag{3b}$$

$$f_x''^{2k} = -\left(\frac{1}{a^k}\right)\mathbf{u}_y''^{ikT}\mathbf{T}_i''^{k}, \tag{3c}$$

$$f_y''^{2k} = \left(\frac{1}{a^k}\right)\mathbf{u}_x''^{ikT}\mathbf{T}_i''^{k}, \tag{3d}$$

$$f_z''^{1k} = \mathbf{u}_z''^{ikT}\mathbf{F}_i''^{k}, \tag{3e}$$

$$f_x''^{3k} = \left(\frac{1}{2\,b^k}\right)\mathbf{u}_z''^{ikT}\mathbf{T}_i''^{k}. \tag{3f}$$

The superscripts 1 and 2 refer to the components at the two end points of the common joint axis $\mathbf{u}_z''$. The distance between these points is the length of the joint a^k. The superscript 3 indicates the components at the end points of the transverse width b^k of the joint geometry acting in directions parallel to the axis $\mathbf{u}_x''$. Naturally, the components $f_z''^{1k}$ and $f_x''^{3k}$ will exist only for revolute and translational joints, respectively.

1. For a cylindrical joint, the resultant normal force at both end points of the joint will contribute to the axial friction force. Considering a parabolic force distribution, the effective normal force is given by $F_n^k = f''^{1k} + f''^{2k}$, where $f''^{lk} = \frac{\pi^3}{24}\sqrt{(f_x''^{lk})^2 + (f_y''^{lk})^2 + \epsilon^2}$, $l = 1, 2$. Additionally, a friction torque will be introduced at each end point; its magnitude is given by $\tau = r_e \,\mathrm{cabs}(F_f''^{lk})$, where r_e is the effective joint radius, and $\mathrm{cabs}(x)$ is a continuous approximation to the absolute function defined as $\sqrt{x^2 + \epsilon^2}$ with some small scalar ϵ.
2. For a revolute joint, the treatment will be similar to a cylindrical joint with the addition of a thrust force in the z''-direction given by $\mathbf{u}_z''^{ikT}\mathbf{F}_i''^{k}$. This thrust force will contribute to a frictional torque about an effective torque radius r_e.
3. For a translational joint, the effective normal force is the absolute sum of all the force components: $F_n^k = \mathrm{cabs}(f_x''^{1k}) + \mathrm{cabs}(f_y''^{1k}) + \mathrm{cabs}(f_x''^{2k}) + \mathrm{cabs}(f_y''^{2k}) + 2\,\mathrm{cabs}(f_x''^{3k})$.

The normal force for any joint type calculated using equations (3a)–(3f) can be plugged in equation (1) to get the scalar frictional force f_{ij} and torque τ_{ij}. Consequently, the generalized friction force vector to be used in the equations of motion can be computed for a pair of bodies i and j by the following equations:

$$\mathbf{Q}_i^{Af} = \begin{bmatrix} \mathbf{A}_i\mathbf{C}_i^k\mathbf{u}_z''^{ik} f_{ij} \\ \mathbf{B}_i^{\mathrm{T}}\mathbf{A}_i\mathbf{C}_i^k\mathbf{u}_z''^{ik} f_{ij} + 2\mathbf{E}_i^{\mathrm{T}}\mathbf{A}_i\mathbf{C}_i^k\mathbf{u}_z''^{ik}\tau_{ij} \end{bmatrix}, \tag{4a}$$

$$\mathbf{Q}_j^{Af} = \begin{bmatrix} -\mathbf{A}_i\mathbf{C}_i^k\mathbf{u}_z''^{ik} f_{ij} \\ -\mathbf{B}_j^{\mathrm{T}}\mathbf{A}_i\mathbf{C}_i^k\mathbf{u}_z''^{ik} f_{ij} - 2\mathbf{E}_j^{\mathrm{T}}\mathbf{A}_i\mathbf{C}_i^k\mathbf{u}_z''^{ik}\tau_{ij} \end{bmatrix}. \tag{4b}$$

These individual generalized friction force vectors can be assembled to give the combined generalized friction force vector of the entire multibody system. Depending upon the assembly of the generalized coordinate vector $\mathbf{q}$, care must be taken to ensure that the frictional forces are appropriately added to the vector of externally applied forces and the frictional torques to the vector of externally applied torques.

3.3 Equations of motion

Scleronomic multibody systems with joint friction are governed by the following second-order index-3 differential-algebraic equations in centroidal generalized coordinates

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{\Phi}_{\mathbf{q}}^{\mathrm{T}}\boldsymbol{\lambda} = \mathbf{Q} + \mathbf{Q}^{Af}, \tag{5}$$

$$\mathbf{\Phi} = \mathbf{0}. \tag{6}$$

The index can be reduced by successively differentiating the algebraic constraints $\mathbf{\Phi}$ to obtain the following index-1 augmented form [75]

$$\begin{bmatrix} \mathbf{M} & \mathbf{\Phi}_{\mathbf{q}}^{\mathrm{T}} \\ \mathbf{\Phi}_{\mathbf{q}} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{q}} \\ \boldsymbol{\lambda} \end{Bmatrix} = \begin{Bmatrix} \mathbf{Q} + \mathbf{Q}^{Af} \\ \mathbf{c} \end{Bmatrix}, \tag{7}$$

where $\mathbf{M}(\mathbf{q}, \boldsymbol{\rho}) \in \mathbb{R}^{n\times n}$ is the generalized mass matrix, $\mathbf{\Phi}(\mathbf{q}, \boldsymbol{\rho})$ and $\boldsymbol{\lambda} \in \mathbb{R}^m$ are the holonomic constraints and the associated Lagrange multipliers, respectively, $\mathbf{q} \in \mathbb{R}^n$ are the generalized coordinates associated with the bodies of the system, $\mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\rho}) \in \mathbb{R}^n$ and

$\mathbf{Q}^{Af}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\lambda}, \boldsymbol{\rho}) \in \mathbb{R}^n$ are the generalized externally applied and friction forces, respectively, $\mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\rho}) \in \mathbb{R}^m$ is the acceleration term equal to $-(\boldsymbol{\Phi}_\mathbf{q}\dot{\mathbf{q}})_\mathbf{q}\dot{\mathbf{q}}$, and $\boldsymbol{\rho} \in \mathbb{R}^p$ is the vector of design and/or control parameters. It is implied in this analysis that the constraint Jacobian $\boldsymbol{\Phi}_\mathbf{q}$ is of full rank. For redundant constraints, the reader is referred to García de Jalón and Gutiérrez-López [30]. The motion of multibody systems with friction is characterized by sharp changes in acceleration. Moreover, the friction is a nonlinear function of Lagrange multipliers, making the equations fully implicit. Additionally, these equations are prone to constraint drift due to enforcement of $\ddot{\boldsymbol{\Phi}}$ instead of $\boldsymbol{\Phi}$. Since these dynamic equations will be differentiated to obtain the TLMs, any errors in the solution of dynamics may further exacerbate the errors in sensitivities. Hence, these equations require some form of implicit integration scheme to contain error propagation with time. Implicit integration schemes help reduce the perturbations introduced in the acceleration field and are more efficient in handling the stiff nature of index-1 formulation.

We can rewrite (7) as an explicit first-order DAE expressed in constant singular mass-matrix form:

$$\begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{q}} \\ \ddot{\mathbf{q}} \\ \dot{\boldsymbol{\lambda}} \end{Bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} & \boldsymbol{\Phi}_\mathbf{q}^\mathrm{T} \\ \mathbf{0} & \boldsymbol{\Phi}_\mathbf{q} & \mathbf{0} \end{bmatrix}^{-1} \begin{Bmatrix} \dot{\mathbf{q}} \\ \mathbf{Q} + \mathbf{Q}^{Af} \\ \mathbf{c} \end{Bmatrix} - \begin{Bmatrix} \mathbf{0} \\ \mathbf{0} \\ \boldsymbol{\lambda} \end{Bmatrix} \tag{8a}$$

$$\implies \bar{\mathbf{M}}\dot{\mathbf{y}} = \mathbf{f}(\mathbf{t}, \mathbf{y}), \tag{8b}$$

where $\mathbf{y} = \left[\mathbf{q}^\mathrm{T}, \dot{\mathbf{q}}^\mathrm{T}, \boldsymbol{\lambda}^\mathrm{T}\right]^\mathrm{T}$. These are linearly implicit differential-algebraic equations [35, 76, 77, 84] that are solved over a time interval $[t_0, t_f]$. Rosenbrock–Wanner (ROW) methods are well known for solving such problems. Another approach for solving these equations involves performing a fixed point iteration every time step [81]. However, fixed point iteration tends to be slow, as it is an additional convergence iteration apart from the implicit integration needed to solve the acceleration terms. A ROW scheme with stage-number s for the problem is defined by

$$(\bar{\mathbf{M}} - h\gamma\mathbf{f}_\mathbf{y})k_i = h\mathbf{f}\left(t_0 + \alpha_i h, \mathbf{y}_p + \sum_{j=1}^{i-1}\alpha_{ij}k_j\right) + h\mathbf{f}_\mathbf{y}\sum_{j=1}^{i-1}\gamma_{ij}k_j + h^2\gamma_i\mathbf{f}_t, \tag{9}$$

$$\mathbf{y}_{p+1} = \mathbf{y}_p + \sum_{i=1}^{s} b_i k_i, \quad i = 1, \ldots, s, \quad p = 0, 1, \ldots, \tag{10}$$

$$\text{where } \mathbf{f}_\mathbf{y} = \frac{\partial\mathbf{f}}{\partial\mathbf{y}}(t_p, \mathbf{y}_p), \tag{11}$$

$$\mathbf{f}_t = \frac{\partial\mathbf{f}}{\partial t}(t_p, \mathbf{y}_p), \tag{12}$$

where h is the time step for the integrator step p. The coefficients of the method are γ, α_{ij}, and γ_{ij}, and b_i are the weights. Moreover, $\alpha_i = \sum_{j=1}^{i-1}\alpha_{ij}$ and $\gamma_i = \gamma + \sum_{j=1}^{i-1}\gamma_{ij}$. It is worth noting that this scheme works well only for index-1 systems, since it guarantees the regularity of the matrix $(\bar{\mathbf{M}} - h\gamma\mathbf{f}_\mathbf{y})$. In addition to the ROW methods described before, a fully implicit Runge–Kutta method (`Radau`) [36] is also suitable for solving equation (8a)–(8b). However, multistep BDF methods like `CVODE`/`FBDF` [42] and `DASSL` [65] were found to be comparatively slow for these problems and also led to failed integration due to high integration errors for some problems. Also, it is possible to solve the equations as fully implicit DAEs in residual form; however, obtaining consistent initial conditions is difficult and may lead to integration failure.

The positions $\mathbf{q}_0$ at the initial time t_0 can be obtained by introducing a set of temporary constraints $\boldsymbol{\Psi} \in \mathbb{R}^{n-m}$ to complete the nonlinear system of constraint equations and make the constraint Jacobian square. Exact generalized coordinates $\mathbf{q}$ can be converged from a given initial estimate using the following Newton–Raphson iterative scheme:

$$\bar{\boldsymbol{\Phi}} = \left\{ \begin{matrix} \boldsymbol{\Phi} \\ \boldsymbol{\Psi} \end{matrix} \right\}_{t_0} = \mathbf{0}, \tag{13a}$$

$$\begin{bmatrix} \boldsymbol{\Phi}_{\mathbf{q}} \\ \boldsymbol{\Psi}_{\mathbf{q}} \end{bmatrix}_{t_0} \Delta \mathbf{q}^i = -\left\{ \begin{matrix} \boldsymbol{\Phi} \\ \boldsymbol{\Psi} \end{matrix} \right\}_{t_0}, \tag{13b}$$

$$\mathbf{q}^{i+1} = \mathbf{q}^i + \Delta \mathbf{q}^i. \tag{13c}$$

For spatial systems, obtaining a reasonably good estimate of $\mathbf{q}$ is a challenge and typically requires a CAD model. A trust-region algorithm [56, 57, 61] from the `NLSolve.jl` package was used to converge all initial estimates to machine precision.

A trivial solution for the initial velocity vector is $\dot{\mathbf{q}} = \mathbf{0}$ since it will always satisfy the velocity constraints $\dot{\boldsymbol{\Phi}} = \boldsymbol{\Phi}_{\mathbf{q}} \dot{\mathbf{q}} = \mathbf{0}$, which implies that the system is at rest. For nonzero initial velocities, the solution has to be a linear combination of the null space vectors of $\boldsymbol{\Phi}_{\mathbf{q}}$ at the initial time such that the velocities of the $n-m$ chosen independent generalized coordinates are satisfied. Alternatively, the velocity constraints $\dot{\boldsymbol{\Phi}}$ can be completed to obtain the generalized velocities [40]. Since nonzero initial velocities lead to nonnegligible frictional forces, it is relatively difficult to obtain initial Lagrange multipliers. In this analysis, we obtain the estimate for initial Lagrange multipliers by ignoring $\mathbf{Q}^{Af}$, temporarily making the equations of motion explicit. Thereafter, an exact solution is obtained through a nonlinear solution of the residual

$$\mathbf{R} = \left\{ \begin{matrix} \ddot{\mathbf{q}} \\ \boldsymbol{\lambda} \end{matrix} \right\} - \begin{bmatrix} \mathbf{M} & \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}} \\ \boldsymbol{\Phi}_{\mathbf{q}} & \mathbf{0} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{Q} + \mathbf{Q}^{Af} \\ \mathbf{c} \end{bmatrix} \to \mathbf{0}. \tag{14}$$

4 Gradient-based optimization

Dynamic optimization aims to find the optimal design or control parameters while minimizing or maximizing a specified objective function subject to the constraints imposed by the system dynamics and parameter/state bounds. This can be mathematically expressed as an initial value bound-constrained optimization problem

$$\min_{\boldsymbol{\rho}} \quad \boldsymbol{\psi}(\mathbf{y}, \boldsymbol{\rho}), \tag{15a}$$

$$\text{such that} \quad \bar{\mathbf{M}}\dot{\mathbf{y}} = \mathbf{f}(\mathbf{y}, \boldsymbol{\rho}, t), \tag{15b}$$

$$\underline{\boldsymbol{\rho}} \le \boldsymbol{\rho} \le \overline{\boldsymbol{\rho}}, \tag{15c}$$

$$\text{and} \quad \mathbf{y}\big|_{t_0} = \mathbf{y}_0, \tag{15d}$$

where $\underline{\boldsymbol{\rho}}, \overline{\boldsymbol{\rho}} \in \mathbb{R}^p$ represent the lower and upper bound vectors, respectively. We will mainly deal with the objective function of the form

$$\boldsymbol{\psi} = \mathbf{w}(\mathbf{y}, \boldsymbol{\rho})\Big|_{t_f} + \int_{t_o}^{t_f} \mathbf{g}(\mathbf{y}, \boldsymbol{\rho})\, \mathrm{d}t. \tag{16}$$

In this context, $\boldsymbol{\psi} \in \mathbb{R}^o$ represents a vector containing "o" objective functions. Additionally, $\mathbf{w} \in \mathbb{R}^o$ corresponds to the pointwise term at the final time, and $\mathbf{g} \in \mathbb{R}^o$ denotes the integrand associated with each objective function. The quadratic cost function shown in equation (17) is a widely applicable specialization of the objective function (16) for control case studies.

$$\min_{\boldsymbol{\rho},\,\mathbf{u}} \int_{t_0}^{t_f} \left[\left(\mathbf{y} - \mathbf{y}_{\mathrm{ref}}\right)^{\mathrm{T}} \mathbf{Q} \left(\mathbf{y} - \mathbf{y}_{\mathrm{ref}}\right) + \mathbf{u}^{\mathrm{T}} \mathbf{R} \mathbf{u} \right] \mathrm{d}t. \tag{17}$$

Here the control $\mathbf{u} = \mathbf{h}(\boldsymbol{\rho}, t)$ contained in a ball of admissible control set $\boldsymbol{\Omega}$ is required to follow a trajectory $\mathbf{y}_{\mathrm{ref}}$, and $\mathbf{Q}$ and $\mathbf{R}$ are positive semidefinite penalty matrices for state error and control, respectively. The motivation behind this choice of objective function is that most optimization problems for dynamic systems can be formulated either as trajectory tracking problems or regulating problems. Time optimal problems of the form

$$\min_{\boldsymbol{\rho}} \quad t_f \tag{18}$$

$$\text{such that} \quad \mathbf{y}\big|_{t_f} = \mathbf{y}_f \tag{19}$$

$$\text{and} \quad \mathbf{y}\big|_{t_0} = \mathbf{y}_0 \tag{20}$$

are out of scope for this research. The gradient of the objective function (16) is given by

$$\begin{aligned} \nabla_{\boldsymbol{\rho}} \boldsymbol{\psi}^{\mathrm{T}} = {} & \left(\mathbf{w}_{\mathbf{q}} \mathbf{q}' + \mathbf{w}_{\dot{\mathbf{q}}} \dot{\mathbf{q}}' + \mathbf{w}_{\boldsymbol{\lambda}} \boldsymbol{\lambda}' + \mathbf{w}_{\boldsymbol{\rho}}\right)\Big|_{t_f} \\ & + \int_{t_o}^{t_f} \left(\mathbf{g}_{\mathbf{q}} \mathbf{q}' + \mathbf{g}_{\dot{\mathbf{q}}} \dot{\mathbf{q}}' + \mathbf{g}_{\boldsymbol{\lambda}} \boldsymbol{\lambda}' + \mathbf{g}_{\boldsymbol{\rho}}\right) \mathrm{d}t, \end{aligned} \tag{21}$$

where the terms $\mathbf{q}', \dot{\mathbf{q}}' \in \mathbb{R}^{n \times p}$ and $\boldsymbol{\lambda}' \in \mathbb{R}^{m \times p}$ are the respective state sensitivities (total derivatives) with respect to the parameters $\boldsymbol{\rho}$. To obtain these sensitivities of the states, we differentiate equation (7) with respect to the parameters $\boldsymbol{\rho}$:

$$\frac{\mathrm{d}\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}}}{\mathrm{d}\boldsymbol{\rho}} + \frac{\mathrm{d}\boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}}(\mathbf{q})\boldsymbol{\lambda}}{\mathrm{d}\boldsymbol{\rho}} = \frac{\mathrm{d}(\mathbf{Q}^{Af}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\lambda}) + \mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}, t))}{\mathrm{d}\boldsymbol{\rho}}, \tag{22a}$$

$$\frac{\mathrm{d}\left(\boldsymbol{\Phi}_{\mathbf{q}} \ddot{\mathbf{q}}\right)}{\mathrm{d}\boldsymbol{\rho}} = \frac{\mathrm{d}\mathbf{c}}{\mathrm{d}\boldsymbol{\rho}}. \tag{22b}$$

If we consider that all terms are dependent on $\boldsymbol{\rho}$, then the total derivatives can be expanded, resulting in the final TLMs:

$$\mathbf{M}\ddot{\mathbf{q}}' + \bar{\mathbf{C}}\dot{\mathbf{q}}' + \bar{\mathbf{K}}\mathbf{q}' + \bar{\mathbf{L}}\boldsymbol{\lambda}' = \bar{\mathbf{Q}}, \quad \text{and} \tag{23a}$$

$$\boldsymbol{\Phi}_{\mathbf{q}} \ddot{\mathbf{q}}' - \mathbf{c}_{\dot{\mathbf{q}}} \dot{\mathbf{q}}' + (\boldsymbol{\Phi}_{\mathbf{qq}} \ddot{\mathbf{q}} - \mathbf{c}_{\mathbf{q}})\mathbf{q}' = \mathbf{c}_{\boldsymbol{\rho}} - \boldsymbol{\Phi}_{\mathbf{q}\boldsymbol{\rho}} \ddot{\mathbf{q}}, \tag{23b}$$

where

$$\bar{\mathbf{C}} = -\mathbf{Q}_{\dot{\mathbf{q}}} - \mathbf{Q}_{\dot{\mathbf{q}}}^{Af}, \tag{24a}$$

$$\bar{\mathbf{K}} = \mathbf{M}_{\mathbf{q}} \ddot{\mathbf{q}} + \boldsymbol{\Phi}_{\mathbf{qq}}^{\mathrm{T}} \boldsymbol{\lambda} - \mathbf{Q}_{\mathbf{q}} - \mathbf{Q}_{\mathbf{q}}^{Af}, \tag{24b}$$

$$\bar{\mathbf{L}} = \boldsymbol{\Phi}_{\mathbf{q}}^{\mathrm{T}} - \mathbf{Q}_{\boldsymbol{\lambda}}^{Af}, \tag{24c}$$

$$\bar{\mathbf{Q}} = \mathbf{Q}_\rho + \mathbf{Q}_\rho^{Af} - \mathbf{M}_\rho \ddot{\mathbf{q}} - \boldsymbol{\Phi}_{\mathbf{q}\rho}^{\mathrm{T}} \boldsymbol{\lambda}, \tag{24d}$$

$$\mathbf{c}_\mathbf{q} = -\dot{\boldsymbol{\Phi}}_{\mathbf{q}\mathbf{q}} \dot{\mathbf{q}}, \tag{24e}$$

$$\mathbf{c}_{\dot{\mathbf{q}}} = -\dot{\boldsymbol{\Phi}}_{\mathbf{q}\dot{\mathbf{q}}} \dot{\mathbf{q}} - \dot{\boldsymbol{\Phi}}_\mathbf{q}, \tag{24f}$$

$$\mathbf{c}_\rho = -\dot{\boldsymbol{\Phi}}_{\mathbf{q}\rho} \dot{\mathbf{q}}. \tag{24g}$$

In equation (24a)–(24g), the frictional force vector $\mathbf{Q}^{Af}$ and the corresponding Jacobians $\mathbf{Q}_\mathbf{q}^{Af}$, $\mathbf{Q}_{\dot{\mathbf{q}}}^{Af}$, $\mathbf{Q}_{\boldsymbol{\lambda}}^{Af}$, and $\mathbf{Q}_\rho^{Af}$ are all dependent on Lagrange multipliers. It is important to note that equations (24a)–(24g) contain several terms such as $\mathbf{M}_\mathbf{q}\ddot{\mathbf{q}}$, $\boldsymbol{\Phi}_{\mathbf{q}\mathbf{q}}^{\mathrm{T}}\boldsymbol{\lambda}$, and so on that are tensor–vector products. These can be computed for any matrix $\mathbf{A} \in \mathbb{R}^{q\times r}$ and any pair of vectors $\mathbf{b} \in \mathbb{R}^r$ and $\mathbf{x} \in \mathbb{R}^s$:

$$\mathbf{A}_\mathbf{x} = \left[\frac{\partial \mathbf{A}}{\partial x_1}, \quad \cdots, \quad \frac{\partial \mathbf{A}}{\partial x_i}, \quad \cdots, \quad \frac{\partial \mathbf{A}}{\partial x_s}\right] \in \mathbb{R}^{q\times r\times s}, \tag{25a}$$

$$\mathbf{A}_\mathbf{x}\mathbf{b} = \left[\frac{\partial \mathbf{A}}{\partial x_1}\mathbf{b}, \quad \cdots, \quad \frac{\partial \mathbf{A}}{\partial x_i}\mathbf{b}, \quad \cdots, \quad \frac{\partial \mathbf{A}}{\partial x_s}\mathbf{b}\right] \in \mathbb{R}^{q\times s}. \tag{25b}$$

However, as differentiation is a linear operation, tensor algebra can be easily avoided by premultiplying the respective matrices with the associated vector before differentiating $\mathbf{A}_\mathbf{x}\mathbf{b} = (\mathbf{A}\mathbf{b})_\mathbf{x}$.

To solve equation (23a)–(23b), a set of $2np$ initial conditions is necessary, represented by position sensitivities $\mathbf{q}'\big|_{t_0} = \mathbf{q}_0'$ and velocity sensitivities $\dot{\mathbf{q}}'\big|_{t_0} = \dot{\mathbf{q}}_0'$. As these initial sensitivities must adhere to the sensitivity constraints at the initial time, they can be determined by solving the following equations:

$$\left.\frac{\mathrm{d}\bar{\boldsymbol{\Phi}}}{\mathrm{d}\boldsymbol{\rho}}\right|_{t_0} = \mathbf{0} \implies \bar{\boldsymbol{\Phi}}_\mathbf{q}\big|_{t_0} \mathbf{q}_0' = -\bar{\boldsymbol{\Phi}}_\rho\big|_{t_0}, \tag{26a}$$

$$\left.\frac{\mathrm{d}\dot{\bar{\boldsymbol{\Phi}}}}{\mathrm{d}\boldsymbol{\rho}}\right|_{t_0} = \mathbf{0} \implies \bar{\boldsymbol{\Phi}}_\mathbf{q}\big|_{t_0} \dot{\mathbf{q}}_0' = -\left(\bar{\boldsymbol{\Phi}}_{\mathbf{q}\mathbf{q}}\mathbf{q}_0' + \bar{\boldsymbol{\Phi}}_{\mathbf{q}\rho}\right)\dot{\mathbf{q}}\big|_{t_0}. \tag{26b}$$

We can observe from equation (26b) that if the system starts at rest, then the velocity sensitivities are also zero. Rearranging equation (23a)–(23b) in an augmented matrix form, we have

$$\begin{bmatrix} \mathbf{M} & \bar{\mathbf{L}} \\ \boldsymbol{\Phi}_\mathbf{q} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{q}}' \\ \boldsymbol{\lambda}' \end{Bmatrix} = \begin{bmatrix} \mathbf{A} \\ \mathbf{B} \end{bmatrix}, \tag{27}$$

where

$$\mathbf{A} = \bar{\mathbf{Q}} - \bar{\mathbf{C}}\dot{\mathbf{q}}' - \bar{\mathbf{K}}\mathbf{q}', \tag{28a}$$

$$\mathbf{B} = \mathbf{c}_\rho - \boldsymbol{\Phi}_{\mathbf{q}\rho}\ddot{\mathbf{q}} + \mathbf{c}_{\dot{\mathbf{q}}}\dot{\mathbf{q}}' - (\boldsymbol{\Phi}_{\mathbf{q}\mathbf{q}}\ddot{\mathbf{q}} - \mathbf{c}_\mathbf{q})\mathbf{q}'. \tag{28b}$$

Equation (27) is an index-1 DAE since the algebraic variable $\boldsymbol{\lambda}'$ does not appear as a differential term anywhere in the sensitivity mass-matrix on the left or the matrices $\mathbf{A}$ and $\mathbf{B}$ on the right. It can be solved for $\ddot{\mathbf{q}}'$ and $\boldsymbol{\lambda}'$ through matrix inversion and integrated to obtain the sensitivities $\dot{\mathbf{q}}'$ and $\mathbf{q}'$. All sensitivities can be simultaneously computed using a solver for differential equations in matrix form or by reshaping the associated matrices at every iteration to make the system compatible with standard vector form ODE solvers. The sensitivities for multibody systems using the Brown–McPhee friction model tend to exhibit abrupt jumps, akin to those seen in the hybrid dynamic systems [81]. Ideally, a stiffness-aware integrator with automatic switching, like `LSODA` [41, 66, 72], should be used for obtaining the sensitivities.

The gradient of the objective function can be calculated with respect to all the parameters once the sensitivities have been obtained and the optimization iterations can be started. The study used quasi-Newton methods, predominantly L-BFGS, which are suitable for bound-constrained optimization problems solved with this methodology. They require only the first-order gradient of the objective function, which is essential for multibody optimizations, since computing the Hessian matrix would be mathematically and computationally infeasible. The algorithm is described briefly for the readers convenience in the context of multibody optimization. For more detail, the reader is referred to [14, 74]. This algorithm approximates the cost function through the following quadratic function:

$$\psi(\boldsymbol{\rho}) \approx \psi(\boldsymbol{\rho}_k) + \nabla\psi(\boldsymbol{\rho}_k)^T(\boldsymbol{\rho} - \boldsymbol{\rho}_k) + \frac{1}{2}(\boldsymbol{\rho} - \boldsymbol{\rho}_k)^T\mathbf{B}_k(\boldsymbol{\rho} - \boldsymbol{\rho}_k), \tag{29}$$

where $\mathbf{B}_k$ is the limited-memory approximation for the Hessian matrix at iteration k. A piecewise linear path is assumed for the design parameters:

$$\boldsymbol{\rho}_{k+1} = \mathbf{P}(\boldsymbol{\rho}_k - t\nabla\psi(\boldsymbol{\rho}_k),\, \underline{\boldsymbol{\rho}},\, \overline{\boldsymbol{\rho}}), \tag{30a}$$

$$\text{where}\, \mathbf{P}(\boldsymbol{\rho}, \underline{\boldsymbol{\rho}}, \overline{\boldsymbol{\rho}}) = \max(\underline{\boldsymbol{\rho}}, \min(\boldsymbol{\rho}, \overline{\boldsymbol{\rho}})). \tag{30b}$$

Equations (30a)–(30b) determine the Cauchy point $\boldsymbol{\rho}^c$, which is the first local minimum of $\psi(\boldsymbol{\rho})$. The variables whose Cauchy point is at the lower or upper bound are held fixed. These comprise the active set $\mathcal{A}(\boldsymbol{\rho}^c)$. The following quadratic problem is considered over the subspace of free variables to calculate an approximate solution $\boldsymbol{\rho}^*_{k+1}$:

$$\min_{\rho}\{\psi(\boldsymbol{\rho}^*) : \rho_i = \rho_i^c,\, \forall i \in \mathcal{A}(\rho^c)\} \tag{31a}$$

$$\text{such that} \quad \underline{\rho}_i < \rho_i < \overline{\rho}_i,\, \forall i \notin \mathcal{A}(\rho^c). \tag{31b}$$

Equation (31a)–(31b) can be solved in two ways. First, the bounds on the free variables can be ignored for optimization, and the solution can be obtained by direct or iterative methods. Then the free variable path can be truncated such that the solution satisfies the bounds. Another approach is to handle the active bounds by Lagrange multipliers. Once an approximate solution $\boldsymbol{\rho}^*_{k+1}$ is obtained, the next iterate $\boldsymbol{\rho}_{k+1}$ can be obtained by backtracking linear search along $\mathbf{d}_k = \boldsymbol{\rho}^*_{k+1} - \boldsymbol{\rho}_k$, which ensures

$$\psi(\boldsymbol{\rho}_{k+1}) \leq \psi(\boldsymbol{\rho}_k) + \alpha_k\nabla\psi(\boldsymbol{\rho}_k)^{\mathrm{T}}\mathbf{d}_k, \tag{32}$$

where α_k is the optimal step size. This process is repeated until convergence by evaluating the gradient at $\boldsymbol{\rho}_{k+1}$ and computing a new limited-memory Hessian approximation $\mathbf{B}_{k+1}$. The optimization package `Optim.jl` and MATLAB® `fminunc` were used to obtain the results of the case studies in this research.

5 Case studies

Multibody optimizations tend to be highly nonconvex due to the large rotations involved in the dynamics. Hence, even though a convex objective function is chosen, the nonlinear dy-

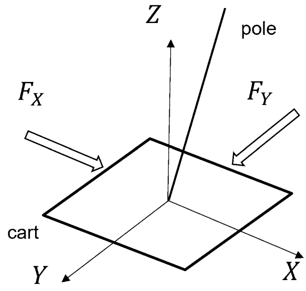


Fig. 2 Schematic of spatial cart-pole

namics makes the solution heavily dependent on the chosen initial condition. This is usually not a concern since gradient-based methods converge fast, allowing the user to experiment with different initial parameter estimates. This section covers three optimization case studies. The first two examples will demonstrate the applicability of this methodology to pure control and pure design problems. Finally, an example of codesign is included to showcase the advantages of the proposed methodology over the classical optimization approach.

5.1 Inverted spatial pendulum

The inverted pendulum, also known as a cart pole problem, is a classic case study in dynamics and control theory, frequently used as a benchmark for testing control strategies. This system is inherently unstable and requires a feedback control loop for stabilization. Interestingly, many real-world systems behave like an inverted pendulum. All bipedal and humanoid robot motions, as well as the motion of bicycles and motorcycles, are similar to that of an inverted pendulum. In this study, we use the spatial variant of the inverted pendulum with a two-dimensional PID controller for stabilization. A schematic for this system is shown in Fig. 2. This additional degree of freedom adds substantial complexity in terms of modeling and control computation. The total number of degrees of freedom (DOFs) for such a system is 5 (2 DOFs for the cart translation along the ground XY plane and 3 rotational DOFs for the pendulum). Friction is nonnegligible at the interface between the cart and ground. This is expected as the weight of the mechanism leads to a high normal reaction at this interface. Friction within the system can be represented through various modeling approaches. The case study is divided into two segments. In the initial part, both the simulation and optimization models incorporate the Brown–McPhee friction model. In the subsequent section, focusing on the importance of friction modeling, the optimization model employs the Brown–McPhee model, whereas the simulation uses the Gonthier friction model.

The goal of this study is to use the methodology presented in this paper for computing the optimal gains for the PID controller, i.e., $\boldsymbol{\rho} = [K_p,\ K_i,\ K_d]$. Manually tuning a PID controller is difficult, especially for nonlinear systems. The methodology described in this paper can be used to convert this optimal feedback control problem into a parameter optimization problem for multibody systems. The optimization is set up to minimize the error in X and Y positions of the cart center $(x_1,\ y_1)$ and the pendulum center $(x_2,\ y_2)$. Thus we obtain

$$e_x = x_2 - x_1 \quad \text{and} \quad e_y = y_2 - y_1. \tag{33}$$

This study has been implemented in MATLAB® using Symbolic Math Toolbox for evaluating Jacobians with respect to design parameters. The objective function chosen for this study is quadratic, with high penalty on the error in states and a comparatively low penalty

Table 2 Inverted pendulum: number of equations

Component	Value
Number of bodies	2
States per body	7
Total differentiable variables for dynamics	$2 \times 7 = 14$
First-order equations of motion	$2 \times 14 = 28$
Degrees of freedom	5
Lagrange multipliers/Constraints	$14 - 5 = 9$
Total dynamic equations	$28 + 9 = 37$
Number of free variables (parameters)	3
Total number of sensitivities	$3 \times 37 = 111$
Total differential-algebraic equations	$37 + 111 = 148$
Total objective function(s)	1
Total objective function gradients	3

on the control. This also has a scaling effect since the control magnitude is much higher than that of the error. The final objective function used for this case study is

$$\psi = (10^5 e_x^2 + 10^5 e_y^2)\Big|_{t_f} + \int_{t_0}^{t_f} 10^5 e_x^2 + 10^5 e_y^2 + u_x^2 + u_y^2 \mathrm{d}t, \tag{34}$$

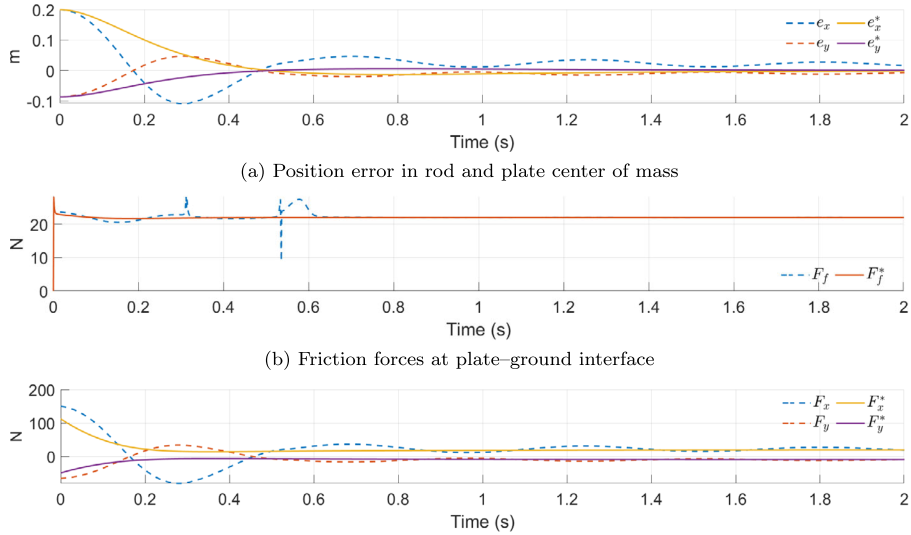
where

$$u_x(t) = K_p e_x + K_i \int_{t_0}^{t} e_x d\mathrm{t} + K_d \frac{\mathrm{d}e_x}{\mathrm{d}t}, \quad \text{and} \tag{35}$$

$$u_y(t) = K_p e_y + K_i \int_{t_0}^{t} e_y d\mathrm{t} + K_d \frac{\mathrm{d}e_y}{\mathrm{d}t}. \tag{36}$$

Hence we penalize both the continuous error and the error at the final time. For gradient-based methods to be effective, it is crucial for users to provide a good initial estimate for the parameters. The chosen cost function has a local minimum if the pendulum hangs vertically below the plate, but this solution is undesirable. To steer the optimization in the desired direction, the initial control should achieve stabilization to some extent. Typically, control saturation constraints are imposed for such optimization studies to consider the limits of the actuation system. However, this paper only deals with bound constraint problems and bounding the gains instead of the computed control would not be appropriate. Hence an unconstrained optimization is performed, and the control saturation limits are imposed indirectly by controlling the state and control penalties in the objective integrand. This approach is suitable for such studies since the time taken for optimization is small and multiple optimization iterations can be easily performed. Table 2 details the number of equations required for this study.

As we can observe in Table 2, a system of 148 differential-algebraic equations must be solved for every optimization iteration along with the additional integrations required to calculate the objective function and its gradient. This is for a relatively simple control case study. Since optimization studies can take several iterations to converge, the computational cost adds up. This highlights the importance of using efficient computation techniques, like sparse-matrix algebra, iterative solvers, SIMD/parallelism, and efficient memory allocation in functions to speed up the process.



(a) Position error in rod and plate center of mass

(b) Friction forces at plate–ground interface

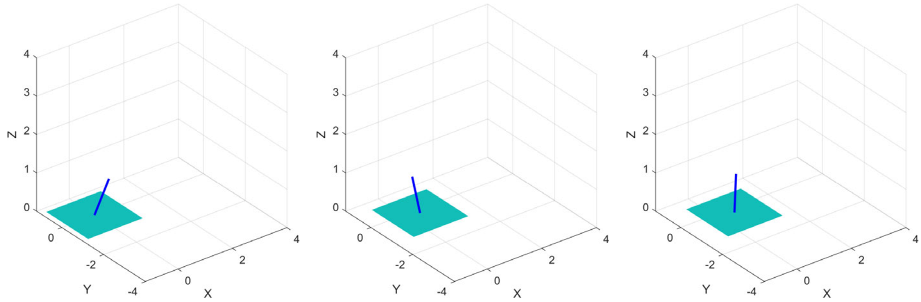
(c) PID control forces acting on the plate

Fig. 3 Inverted pendulum dynamics: initial vs. optimal controller parameters

Figure 3 shows the dynamic response of the system before and after optimization. The PID gain initial estimates were [750, 200, 10], which converged post-optimization to [563.11, 626.57, 90.68]. Figure 3a shows the error dynamics for the system with the initial control versus the optimized control. As we can see, the response is significantly better post-optimization. Figure 3b shows the friction at the plate–ground interface. The preoptimization friction switched from dynamic friction to stiction and back at 0.3 sec and 0.55 sec notably. This is a consequence of aggressive control overcompensating for the stabilization error. The cart keeps changing its direction of motion causing the friction to behave in a way that opposes this change. The friction force is constant for the optimized control as it provides the optimal compensation to achieve stabilization. Figure 3c shows the control forces before and after optimization. The maximum control effort and energy used post-optimization is much lower, however the control stabilizes the pole in less than 1/5th of the time taken by the unoptimized controller.

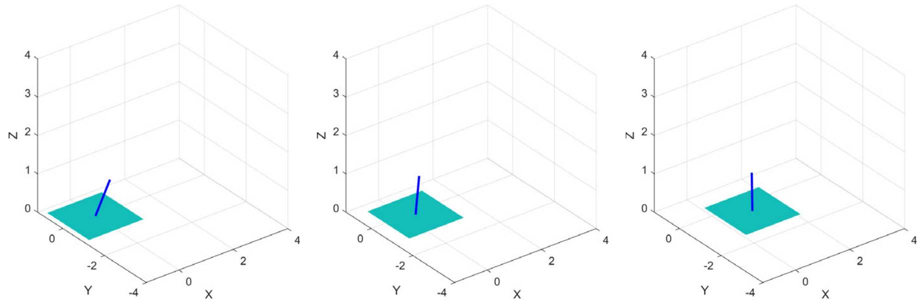
This can be seen graphically in Figs. 4 and 5, which show the system state at various simulation times before and after optimization. The difference in the stabilization performance can be clearly observed through the position of the pole against the timestamps. Figure 6 shows the convergence of the optimization algorithm. The top subplot is the value of the objective function plotted against the optimization iterations. The optimization process achieved greater than 60% reduction in the objective function.

Before transitioning to a different case study, it is vital to understand the importance of modeling friction, especially for control problems. Let us suppose that we decide to ignore the friction after making the observation that its magnitude is less than 10% of the maximum actuator force. The optimization of control parameters is then carried out without any consideration of friction, and the control parameters obtained are used in simulation with friction. Figure 7 compares the stabilization error of the system under such assumptions against previously obtained control parameters where friction was considered during optimization.



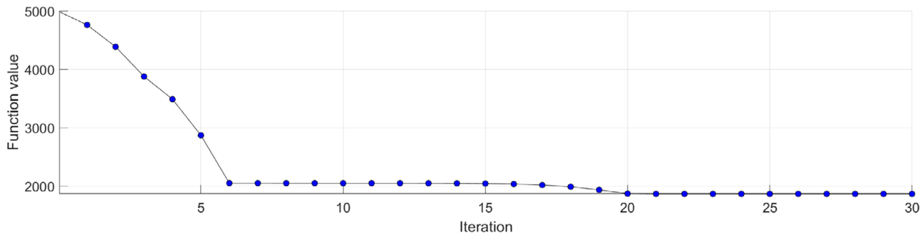
(a) t = 0 (b) t = 300 ms (c) t = 1100 ms

Fig. 4 Inverted pendulum stabilization: initial PID control parameters

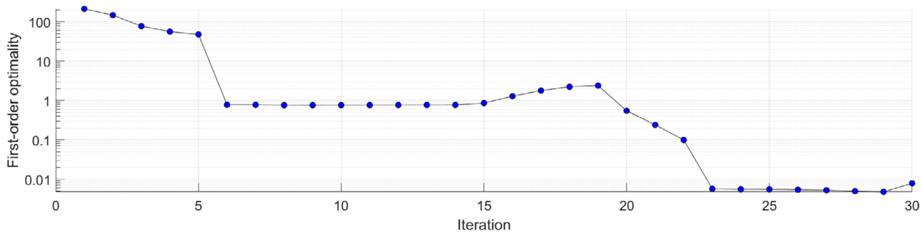


(a) t = 0 ms (b) t = 300 ms (c) t = 1100 ms

Fig. 5 Inverted pendulum stabilization: optimal PID control parameters

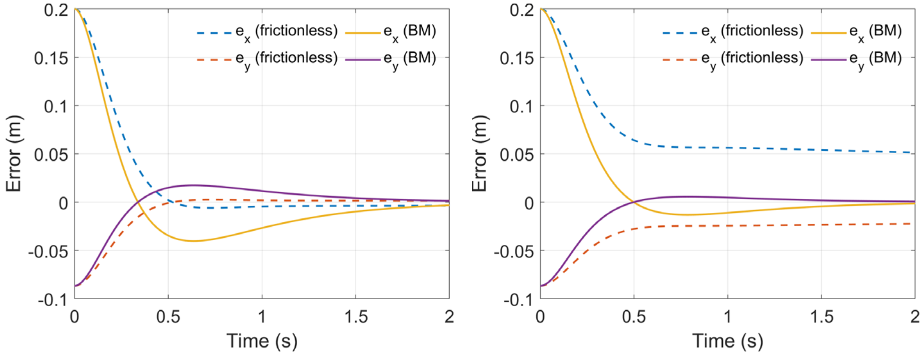


(a) Convergence of objective function for inverted pendulum stabilization



(b) First-order optimality of objective gradient for inverted pendulum stabilization

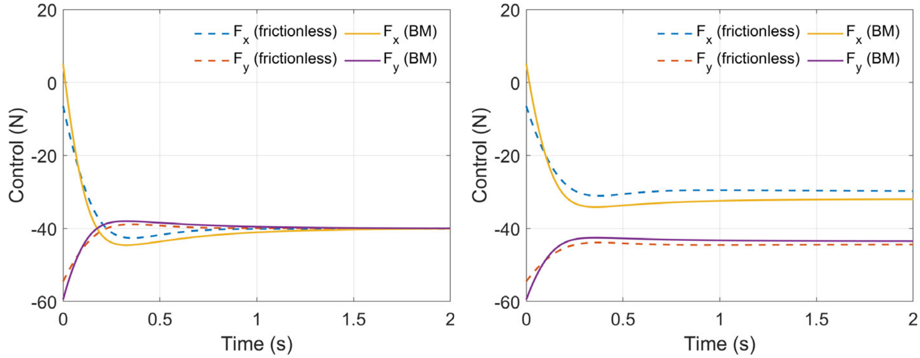
Fig. 6 Inverted pendulum optimization convergence



(a) Frictionless Surface

(b) Surface with Gonthier Friction

Fig. 7 Friction versus frictionless optimization error dynamics



(a) Frictionless surface

(b) Surface with Gonthier friction

Fig. 8 Friction versus frictionless optimization control forces

The *dotted* lines represent the optimization model where friction was not considered, and the *solid* lines represent the model with friction. If the real-world friction was zero, in other words, the surface on which the plate moves is frictionless, the model without friction performs much better in terms of settling time. Figure 8 plots the actuator forces required for stabilization of these models. The model where friction was not considered during optimization achieves stabilization with lower control force magnitudes in comparison to the model where friction was considered during optimization.

Figures 7(b) and 8(b) plot the dynamics for a *real-world* setting, where the friction has been modeled using the Gonthier friction model. It also employs different values for static and dynamic friction coefficients in comparison to the Brown–McPhee model utilized for optimization ([0.5, 0.4] instead of [0.4, 0.3]). It is evident that the model where friction was not considered during optimization has worse performance than the one where friction was considered. The optimization process for the model without friction converged to a local optimum [419.42, 30.30, 65.95] instead of the previous optimal point [563.11, 626.57, 90.68]. The optimization ignored the integral gain of the PID controller, rendering it effectively a PD controller. Thereby we see an offset error that does not seem to go away. This occurred because the optimization aimed to minimize control effort without knowledge of the need

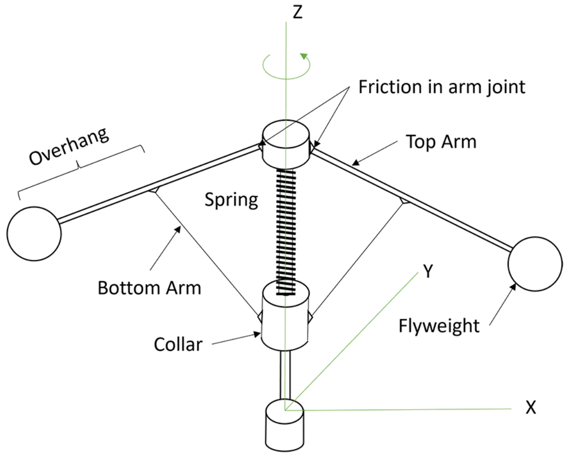


Fig. 9 Flyball governor schematic

for integral gain to counteract friction. As a result, it converged to an optimal point with negligible integral gain. The friction force, although comparatively small in magnitude to the control forces, has a significant influence on dynamics and thereby cannot be ignored. This study also justifies that the friction model used in the optimization model need not be an exact representation of the friction encountered in real world.

5.2 Centrifugal governor mechanism

A centrifugal governor, also known as a flyball governor, is typically used to maintain the speed of a combustion engine by regulating the flow of fuel or working fluid. Several commercial applications such as diesel generators and lawn mowers use centrifugal governors. Designing governors for target engine speeds is well understood, making this mechanism an excellent benchmark example for testing the optimization methodology. This mechanism is a type of servo-mechanical proportional controller, and its analysis as a dynamic system is not trivial. Figure 9 shows an example of a type of flyball governor.

The flyweights attached to the arms are pivoted on a rotating shaft. The collar is attached to the top arm via links and is constrained to translate along the shaft. The entire mechanism rotates with the same angular velocity, say ω, which is proportional to the engine speed. As ω decreases due to an increase in engine load, the centrifugal forces on the weights decreases causing them to move inward, thereby decreasing the collar's height and vice versa. The collar can be connected to the throttle valve of the engine to control the fuel flow and thereby the engine speed. This provides the said governing action for the engine. A key component of such governors is the translational spring damper (TSD), which opposes the motion of the collar to provide a damped dynamic response. For the purpose of this study, the TSD stiffness $k_s = 1000$ N/m and damping of $c_s = 50.0$ N s/m has been used. The servo-mechanism torque or torque at the pillar τ_{pillar} has been modeled using a linear proportional controller governed by the equation

$$\tau_{\text{pillar}} = K_p(h_0 - z_2), \tag{37}$$

where $K_p = 200.0$ N m/m is the proportional gain in torque, $h_0 = 0.1$ m is some predefined height of the collar where the torque would be zero, and z_2 is the dynamic height of the collar obtained in simulation. The system is highly nonlinear, especially if friction effects

Table 3 Flyball governor: number of equations

Component	Value
Number of bodies	6
States per body	7
Total differentiable variables for dynamics	$6 \times 7 = 42$
First-order equations of motion	$2 \times 42 = 84$
Degrees of freedom	2
Lagrange multipliers/Constraints	$42 - 2 = 40$
Total dynamic equations	$84 + 40 = 124$
Number of free variables (parameters)	7
Total number of sensitivities	$7 \times 124 = 868$
Total differential-algebraic equations	$124 + 868 = 992$
Total objective function(s)	1
Total objective function gradients	7

are considered. Therefore it is a good benchmark problem to test the methodology. Such mechanisms are severely neglected in terms of maintenance. Hence it is imperative to ensure that the mechanism functions are as desired even if improper maintenance, such as lack of adequate lubrication, leads to friction in this mechanism.

In terms of modeling, the system contains six bodies (two top links, two bottom links, a vertical pillar which rotates about the Z axis, and a sliding collar). This mechanism is axisymmetric and can also be modeled with four bodies by adding the symmetrical centrifugal forces in the generalized force vector. Table 3 details the number of equations required for this study. The friction in this system has been modeled in the revolute joints of the top arms, which hold the flyweights. This joint is bound to experience high constraint and inertial forces due to the rotation of the governor. Hence, due to improper maintenance, this joint will experience increased frictional torque. The friction at the collar-pillar translational joint will be low, since the centrifugal forces cancel out, leading to negligible constraint reaction forces at that joint. The objective of this particular example is to modify the governor design to achieve a certain desired stable speed, in the presence of friction:

$$\min_{\rho} \psi = \int_{t_0}^{t_f} (\omega_z - 15.0)^2 \mathrm{d}t, \tag{38a}$$

$$\text{where} \quad \boldsymbol{\omega} = \left[\,\omega_x,\, \omega_y,\, \omega_z\,\right]^{\mathrm{T}}. \tag{38b}$$

The angular velocity for any body i is purely a function of its Euler parameters $\mathbf{p}_i$. We have $\boldsymbol{\omega}_i = \mathbf{E}(\mathbf{p}_i)\dot{\mathbf{p}}_i$. The objective function can be expressed mathematically by equation (38a).

The design parameters for this case study are shown in Table 4. As we can see in Fig. 9, the bottom arm of the governor does not connect with the top arm at the vertex, resulting in an overhang. This gives the flyweights leverage to lift the collar against the spring force. The ratio of the overhang to the total length of the top arm is another design parameter for the system. Parameters like pillar height and collar outer radius may not have a substantial effect on the optimization.

Figure 10 shows the initial and optimized trajectory of the governor speed. The optimization achieves the target speed of 15 rad/s. Almost all parameters have decreased in magnitude, except for the bottom arm length. The correlation of the top and bottom arm length to pendulum speed is peculiar. Moving the centrifugal masses outward should decrease the

Table 4 Flyball governor parameter values

Parameter	Description	Initial est.	Bounds	Optima
ρ_1	Top arm length	0.135 m	[0.1 m, 0.15 m]	0.1247 m
ρ_2	Bottom arm length	0.08 m	[0.06 m, 0.1 m]	0.0853 m
ρ_3	Overhang ratio	0.3	[0.2, 0.4]	0.20
ρ_4	Top arm to pillar offset	0.025 m	[0.02 m, 0.03 m]	0.02 m
ρ_5	Collar outer radius	0.025 m	[0.02 m, 0.03 m]	0.02 m
ρ_6	Pillar height	0.225 m	[0.2 m, 0.25 m]	0.2 m
ρ_7	Flyweight mass	0.03 kg	[0.02, 0.04]	0.0242 kg

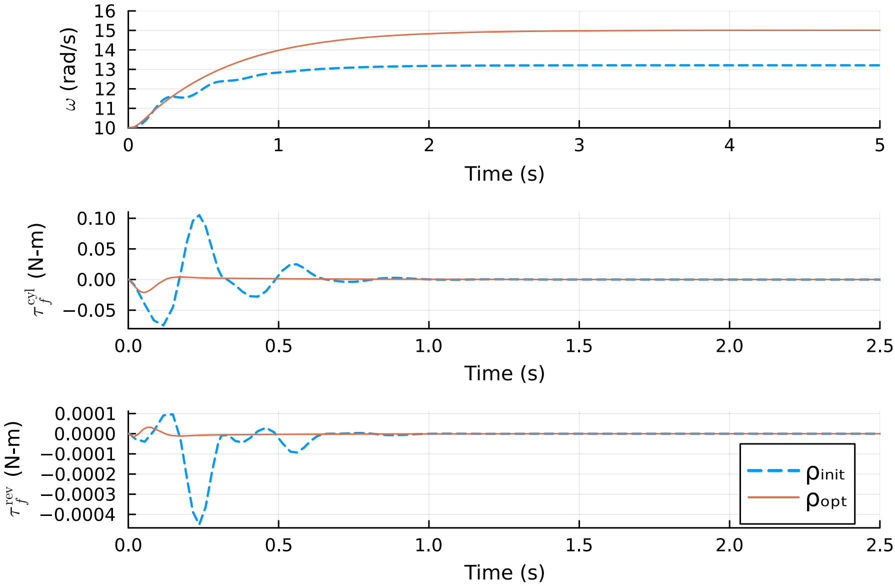


Fig. 10 Flyball governor dynamic plots

target speed since the same centrifugal force can be achieved at a lower speed. However, changing the top arm length and overhang ratio changes the geometry of the mechanism, specifically the joint location for the top and bottom arms. Hence, to maintain this constraint, the length of the bottom arm increased by a small amount even though the top arm length decreased in the optimal design.

Figure 11 shows the optimization converging to a local minimum. It is important to mention that the performance of this particular optimization case study was far from ideal due to the high number of differential equations and suboptimal memory management. This study is intended as a proof of concept of the optimization methodology, and improvements to the execution speed and computational efficiency are planned in the near future.

5.3 Spatial slider-crank mechanism

The slider-crank mechanism is a multibody system that converts rotational motion at the crank into oscillating translational motion at the slider. This mechanism is ubiquitous and is

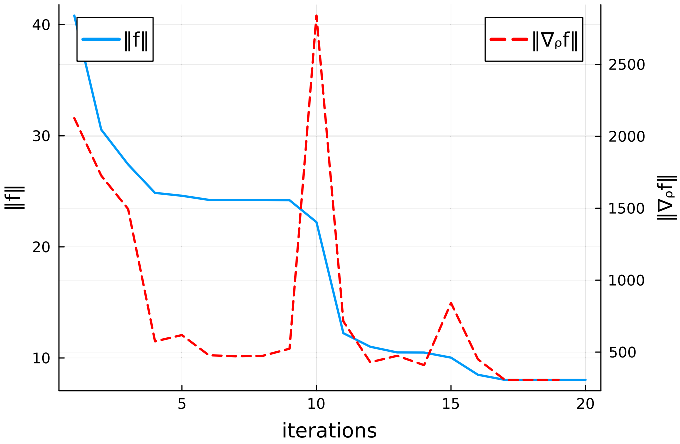


Fig. 11 Flyball governor convergence

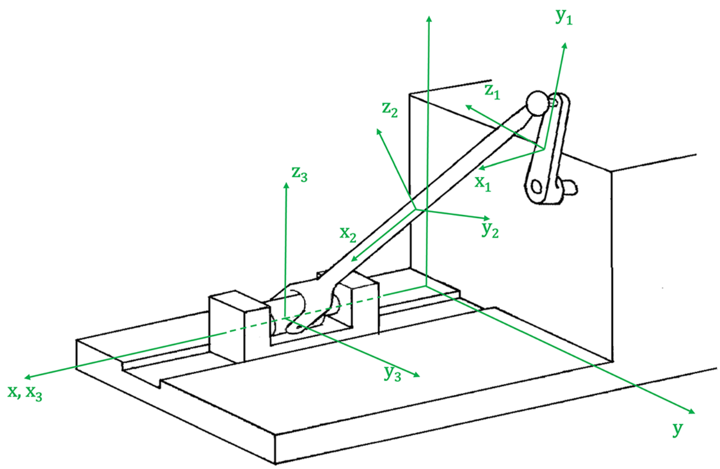


Fig. 12 Slider-crank mechanism schematic

used in combustion engines and various manufacturing processes, such as shaper machines, sheet metal punching machines, and shearing machines. For the purpose of this study, a spatial version of the mechanism is used, as shown in Fig. 12. The model is derived from Haug (1989) [38], which contains the kinematic analysis of the mechanism.

In a previous paper [81], the sensitivity analysis of this mechanism was presented, and the dynamic response was benchmarked using two friction models. It was also demonstrated that the sensitivities of multibody systems with Brown–McPhee friction behave like hybrid dynamic systems by displaying abrupt jumps. This paper will build upon the aforementioned study and illustrate how the optimization methodology presented herein can be applied for codesign optimization. Table 5 highlights the number of equations to be solved for every optimization iteration.

The objective of this study is to design the system and control parameters such that the crank spins at a given constant rotational velocity through a proportional control mechanism.

Table 5 Slider-crank: number of equations

Component	Value
Number of bodies	3
States per body	7
Total differentiable variables for dynamics	$3 \times 7 = 21$
First-order equations of motion	$2 \times 21 = 42$
Degrees of freedom	1
Lagrange multipliers/Constraints	$21 - 1 = 20$
Total dynamic equations	$42 + 20 = 62$
Number of free variables (parameters)	5
Total number of sensitivities	$5 \times 62 = 310$
Total differential-algebraic equations	$62 + 310 = 372$
Total objective function(s)	1
Total objective function gradients	5

Table 6 Slider-crank parameter values

Parameter	Description	Initial	Bounds	Optima
ρ_1	Crank length	0.08 m	[0.06 m, 0.1 m]	0.06 m
ρ_2	Connecting rod length	0.3 m	[0.2 m, 0.4 m]	0.2 m
ρ_3	Slider length	0.05	[0.04 m, 0.06 m]	0.0467 m
ρ_4	Slider width	0.025 m	[0.020 m, 0.03 m]	0.02 m
ρ_5	Proportional gain	1.0	[–5.0, 5.0]	1.0495

Friction at the slider-ground interface is non-negligible. Table 6 details the design and control parameters used for this study. The control torque on the crank τ_{crank} can be represented by the equation

$$\tau_{\text{crank}} = (\omega - \omega_0) K_p, \tag{39}$$

where ω is the current dynamic angular velocity of the crank, $\omega_0 = -10.0$ rad/s is some predefined speed for the crank rotation, and K_p is the tunable proportional controller for crank torsional actuator. The objective function used in the study is

$$\psi = \int_{t_0}^{t_f} \left[(\omega - \omega_0)^2 + 0.01 \tau_{\text{crank}}^2\right] \mathrm{d}t = \int_{t_0}^{t_f} \left[(\omega - \omega_0)^2 (1 + 0.01 K_p^2)\right] \mathrm{d}t. \tag{40}$$

Figure 13 shows the dynamic plots before and after optimization. We can see that the crank is continuously rotating at approximately 10 rad/s with the optimized parameters. The maximum magnitude of control required is also lower. This is a typical advantage of codesign optimization. It will often be the case that the control effort can be substantially reduced if appropriate design is chosen as seen in this example.

Figure 14 shows the friction and effective normal reaction at the translational joint. As we can observe, the initial response of the system was highly oscillatory, which was due to high jerks when the crank was at the bottom most position.

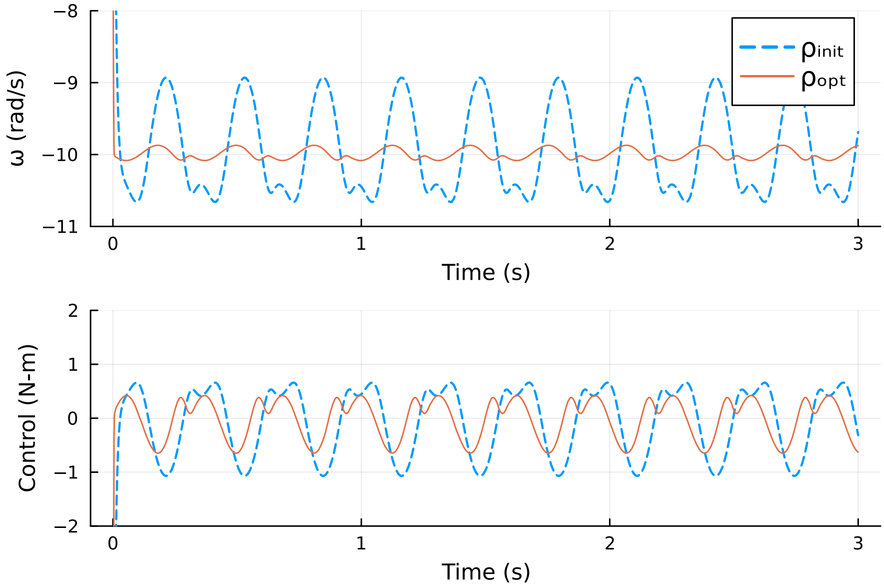


Fig. 13 Slider-crank dynamics

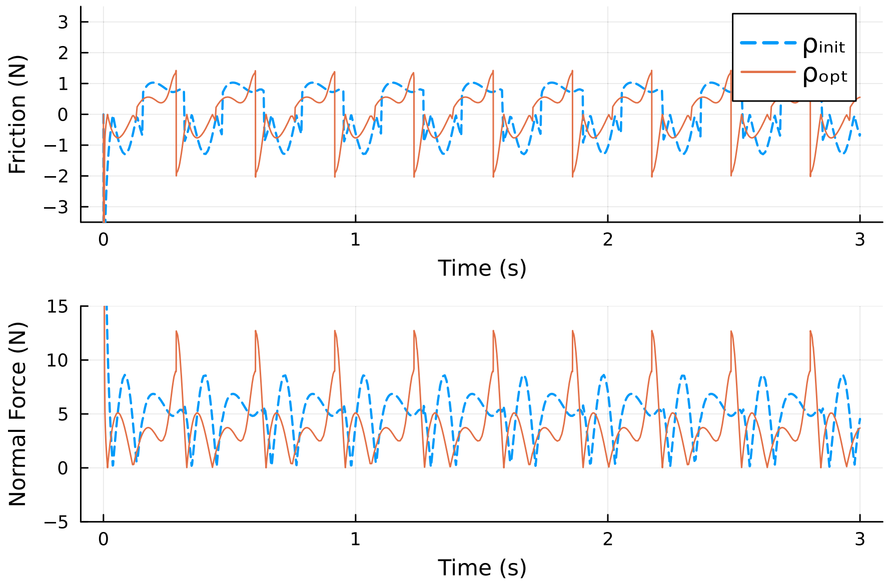


Fig. 14 Slider-crank normal and friction forces

Figure 15 shows the convergence of the optimization problem. The objective function has decreased by about 66% in comparison to the initial estimate of design and control parameters.

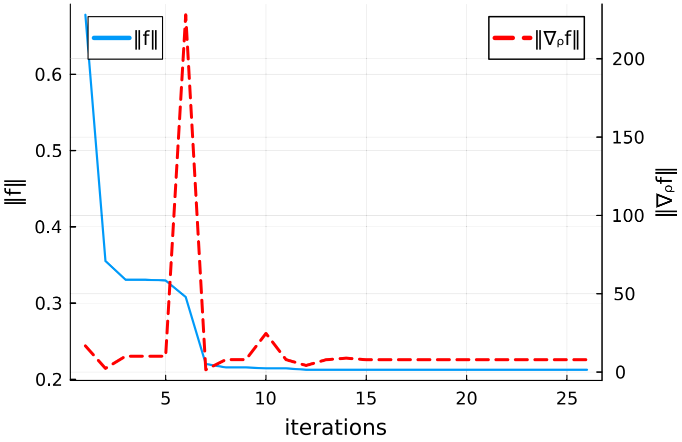


Fig. 15 Slider-crank convergence

6 Conclusions

This paper covered the direct sensitivity methodology for optimization of multibody systems with friction. It was demonstrated how systems with friction can be practically modeled using singular mass-matrix form of differential-algebraic equations that are compatible with standard open-source solvers. Sensitivity analysis through direct differentiation of such dynamic equations can be solved through explicit solvers with adaptive stiffness control. The examples examined in this study underscore the versatility of this methodology in both control and design parameter optimization while demonstrating the advantages of codesign. The methodology is also applicable to control shaping and dynamic estimation problems; however, adjoint sensitivity approach would be preferable in this case due to the large number of parameters. As a future scope of this work, more efficient sparse-matrix implementations need to be executed with nonallocating functions to further speed up the execution time. Additionally, general adjoint sensitivity approaches for optimal control based on calculus of variations plan to be included. An extension of the work for sensitivity analysis and optimization of multibody systems with joint clearances is being developed in parallel.

Nomenclature

nb	Number of bodies in the system.
n	Number of generalized coordinates. $7nb$ for reference point coordinates with Euler parameters.
$\mathbf{q} \in \mathbb{R}^n$	Vector of generalized coordinates.
$\boldsymbol{\lambda} \in \mathbb{R}^m$	Vector of Lagrange multipliers.
$\boldsymbol{\rho} \in \mathbb{R}^p$	Vector of system parameters $[\rho_1, \ldots, \rho_p]^{\mathrm{T}}$.
$\mathbf{y}_{\mathbf{x}}$	Jacobian formed by partial derivatives $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$.
$\dot{x}, \ddot{x}$	Time derivatives $\frac{\mathrm{d}x}{\mathrm{d}t}$ and $\frac{\mathrm{d}^2x}{\mathrm{d}t^2}$ respectively.
$\boldsymbol{\Phi}(\mathbf{q}, \boldsymbol{\rho}) \in \mathbb{R}^m$	Vector of m holonomic constraints.
$\boldsymbol{\Phi}_{\mathbf{q}} \in \mathbb{R}^{m\times n}$	The constraint vector Jacobian. Must be of rank m.
$\mathbf{M}(\mathbf{q}, \boldsymbol{\rho}) \in \mathbb{R}^{n\times n}$	Generalized mass matrix for multibody systems.
$\mathbf{Q}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\rho}) \in \mathbb{R}^n$	Vector of external generalized forces and torques.
$\mathbf{Q}^{Af}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\lambda}, \boldsymbol{\rho}) \in \mathbb{R}^n$	Vector of generalized frictional forces.

v	Magnitude of relative sliding velocity.
v_t	Magnitude of transition velocity.
$\boldsymbol{\mu}$	$[\mu_d, \quad \mu_s]^{\mathrm{T}}$, wherein μ_d and μ_s represent coefficients for dynamic and static friction, respectively.
$F_n(\mathbf{q}, \boldsymbol{\lambda}, \boldsymbol{\rho})$	Magnitude of normal force of contact.
F_f	Magnitude of friction force.
$\mathbf{r}_i \in \mathbb{R}^3$	Position vector in global frame for the origin of i^{th} body-fixed reference frame.
$\mathbf{u}_x, \mathbf{u}_y, \mathbf{u}_z \in \mathbb{R}^3$	Unit axis representing a coodinate system.
$(\cdot), (\cdot)', (\cdot)''$	Physical entities represented in the ground, body-fixed, and joint reference frames respectively.
$\tilde{\mathbf{x}} \in \mathbb{R}^{3\times 3}$	The matrix $[0, -x_3, x_2; x_3, 0, -x_1; -x_2, x_1, 0]$ formed using the elements of a 3-vector $\mathbf{x} = [x_1; x_2; x_3]$. For any other 3-vector $\mathbf{y}$, we have $\tilde{\mathbf{x}}\mathbf{y} = \mathbf{x} \times \mathbf{y}$.
$\mathbf{p}_i \in \mathbb{R}^4$	The Euler parameter vector for the i^{th} body-fixed reference frame is expressed as $[e_0 \quad \mathbf{e}^{\mathrm{T}}]^{\mathrm{T}}$, where $e_0 = \cos(\chi/2)$, and $\mathbf{e} = \mathbf{u}\sin(\chi/2)$ for a given axis-angle rotation $\mathbf{u}$ and χ.
$\mathbf{s}_i' \in \mathbb{R}^3$	Position vector for joint location in the local i^{th} body-fixed reference frame.
$\mathbf{A}_i \in \mathbb{R}^{3\times 3}$	The rotation matrix in terms of Euler parameters, which denotes the transformation from the i^{th} body-fixed coordinate system to the global coordinate system, is given by $\mathbf{A}(\mathbf{p}) = (e_0^2 - \mathbf{e}^{\mathrm{T}}\mathbf{e})\mathbf{I} + 2\mathbf{e}\mathbf{e}^{\mathrm{T}} + 2e_0\tilde{\mathbf{e}}$.
$\mathbf{B}_i \in \mathbb{R}^{3\times 4}$	$2\left[\left(e_0\mathbf{I} + \tilde{\mathbf{e}}\right)\mathbf{s}_i' \quad \mathbf{e}\mathbf{s}_i'^{\mathrm{T}} - \left(e_0\mathbf{I} + \tilde{\mathbf{e}}\right)\tilde{\mathbf{s}}_i'\right]$.
$\mathbf{E}_i \in \mathbb{R}^{3\times 4}$	$\left[-\mathbf{e} \quad \tilde{\mathbf{e}} + e_0\mathbf{I}\right]$.
$\mathbf{G}_i \in \mathbb{R}^{3\times 4}$	$\left[-\mathbf{e} \quad -\tilde{\mathbf{e}} + e_0\mathbf{I}\right]$.
$\mathbf{C}_i \in \mathbb{R}^{3\times 3}$	The rotation matrix that signifies a transformation from the joint definition frame to the i^{th} body-fixed frame.
h	Time step.

Acknowledgements The authors would like to thank Andrey A. Popov and Reid Gomillion for their help with formulation of the dynamics and suggesting appropriate differential equation solvers. The authors would also like to thank Prof. Edward J. Haug for his feedback on this work.

Author contributions AV made substantial contributions to the conception and drafting of the work, analysis and interpretation of data, and the creation of new software used in the work; AS and DD contributed significantly to the numerical aspects of the software used in the work and reviewed the implementation for computational efficiency; AS and CS provided crucial feedback for improving the impact of the work by suggesting good case studies; AS, CS, and DD provided critical revisions regarding technical content, language flow and grammar; All authors agree to be accountable for all aspects of the work in ensuring that questions related to the accuracy or integrity of any part of the work are appropriately investigated and resolved.

Data Availability No datasets were generated or analysed during the current study.

Declarations

Competing interests The authors declare no competing interests.

References

1. Åkesson, J., Årzén, K.E., Gäfvert, M., Bergdahl, T., Tummescheit, H.: Modeling and optimization with Optimica and JModelica.org—languages and tools for solving large-scale dynamic optimization problems. Comput. Chem. Eng. **34**(11), 1737–1749 (2010). https://doi.org/10.1016/j.compchemeng.2009.11.011
2. Alappat, C., Basermann, A., Bishop, A.R., Fehske, H., Hager, G., Schenk, O., Thies, J., Wellein, G.: A recursive algebraic coloring technique for hardware-efficient symmetric sparse matrix-vector multiplication. ACM Trans. Parallel Comput. **7**(3), 19 (2020). https://doi.org/10.1145/3399732
3. Alavi, A., Dolatabadi, M., Mashhadi, J., Noroozinejad Farsangi, E.: Simultaneous optimization approach for combined control–structural design versus the conventional sequential optimization method. Struct. Multidiscip. Optim. **63**(3), 1367–1383 (2021). https://doi.org/10.1007/s00158-020-02765-3
4. Allison, J.T., Han, Z.: Co-design of an active suspension using simultaneous dynamic optimization. In: Design Automation Conference (2014)
5. Andersson, J.A.E., Gillis, J., Horn, G., Rawlings, J.B., Diehl, M.: Casadi: a software framework for nonlinear optimization and optimal control. Math. Program. Comput. **11**(1), 1–36 (2019). https://doi.org/10.1007/s12532-018-0139-4
6. Baker, A.H., Jessup, E.R., Manteuffel, T.: A technique for accelerating the convergence of restarted GMRES. SIAM J. Matrix Anal. Appl. **26**(4), 962–984 (2005). https://doi.org/10.1137/S0895479803422014
7. Baumgarte, J.: Stabilization of constraints and integrals of motion in dynamical systems. Comput. Methods Appl. Mech. Eng. **1**(1), 1–16 (1972). https://www.sciencedirect.com/science/article/pii/0045782572900187
8. Baydin, A.G., Pearlmutter, B.A., Radul, A.A., Siskind, J.M.: Automatic differentiation in machine learning: a survey (2018)
9. Bischof, C., Carle, A., Corliss, G., Griewank, A., Hovland, P.: ADIFOR–generating derivative codes from Fortran programs. Sci. Program. **1**(1), 11–29 (1992). https://doi.org/10.1155/1992/717832
10. Blanchard, E., Sandu, A., Sandu, C.: Parameter estimation for mechanical systems via an explicit representation of uncertainty. Eng. Comput. **26**(5), 541–569 (2009). https://doi.org/10.1108/02644400910970185
11. Blanchard, E.D., Sandu, A., Sandu, C.: A polynomial chaos-based Kalman filter approach for parameter estimation of mechanical systems. J. Dyn. Syst., Meas. Control, Trans. ASME **132**(6), 1–18 (2010). https://doi.org/10.1115/1.4002481
12. Blanchard, E.D., Sandu, A., Sandu, C.: Polynomial chaos-based parameter estimation methods applied to a vehicle system. J. Multi-Body Dyn. **224**(1), 59–81 (2010). https://doi.org/10.1243/14644193JMBD204
13. Brown, P., McPhee, J.: A continuous velocity-based friction model for dynamics and control with physically meaningful parameters. J. Comput. Nonlinear Dyn. **11**(5), 1–6 (2016). https://doi.org/10.1115/1.4033658
14. Byrd, R.H., Lu, P., Nocedal, J., Zhu, C.: A limited memory algorithm for bound constrained optimization. SIAM J. Sci. Comput. **16**(5), 1190–1208 (1995). https://doi.org/10.1137/0916069
15. Callejo, A.: Dynamic response optimization of vehicles through efficient multibody formulations and automatic differentiation techniques. Ph.D. thesis, E.T.S.I. Industriales, Universidad Politénica de Madrid (2013)
16. Callejo, A., Dopico, D.: Direct sensitivity analysis of multibody systems: a vehicle dynamics benchmark. J. Comput. Nonlinear Dyn. **14**(2), 1–9 (2019). https://doi.org/10.1115/1.4041960
17. Canudas de Wit, C., Olsson, H., Astrom, K., Lischinsky, P.: A new model for control of systems with friction. IEEE Trans. Autom. Control **40**(3), 419–425 (1995). https://doi.org/10.1109/9.376053
18. Chang, C.O., Nikravesh, P.E.: Optimal design of mechanical systems with constraint violation stabilization method. J. Mech. Transm. Autom. Des. **107**(4), 493–498 (1985). https://doi.org/10.1115/1.3260751
19. Collins, S., Ruina, A., Tedrake, R., Wisse, M.: Efficient bipedal robots based on passive-dynamic walkers. Science **307**(5712), 1082–1085 (2005). http://www.jstor.org/stable/3840156
20. Corner, S., Sandu, C., Sandu, A.: Modeling and sensitivity analysis methodology for hybrid dynamical system. Nonlinear Anal. Hybrid Syst. **31**, 19–40 (2019). https://doi.org/10.1016/j.nahs.2018.07.003

21. Corner, S., Sandu, A., Sandu, C.: Adjoint sensitivity analysis of hybrid multibody dynamical systems. Multibody Syst. Dyn. **49**(4), 395–420 (2020). https://doi.org/10.1007/s11044-020-09726-0
22. Corral, E., García, M.G., Castejon, C., Meneses, J., Gismeros, R.: Dynamic modeling of the dissipative contact and friction forces of a passive biped-walking robot. Appl. Sci. **10**(7), 2342 (2020). https://doi.org/10.3390/app10072342
23. Dopico, D., Zhu, Y., Sandu, A., Sandu, C.: Direct and adjoint sensitivity analysis of ordinary differential equation multibody formulations. J. Comput. Nonlinear Dyn. (2015). https://doi.org/10.1115/1.4026492
24. Elmqvist, H., Otter, M., Neumayr, A., Hippmann, G.: Modia – equation based modeling and domain specific algorithms. In: Sjölund, M., Buffoni, L., Pop, A., Ochel, L. (eds.) 14th International Modelica Conference, Linköping Electronic Conference Proceedings 181, pp. 73–86. Linköping University Electronic Press, Linköping (2021). https://elib.dlr.de/144872/
25. Feehery, W.F., Tolsma, J.E., Barton, P.I.: Efficient sensitivity analysis of large-scale differential-algebraic systems. Appl. Numer. Math. **25**(1), 41–54 (1997). https://doi.org/10.1016/S0168-9274(97)00050-0
26. Feng, S., Whitman, E., Xinjilefu, X., Atkeson, C.G.: Optimization based full body control for the atlas robot. In: 2014 IEEE-RAS International Conference on Humanoid Robots, pp. 120–127 (2014). https://doi.org/10.1109/HUMANOIDS.2014.7041347
27. Flores, P., Ambrósio, J., Claro, J.P.: Dynamic analysis for planar multibody mechanical systems with lubricated joints. Multibody Syst. Dyn. **12**(1), 47–74 (2004). https://doi.org/10.1023/B:MUBO.0000042901.74498.3a
28. Flores, P., Ambrósio, J., Claro, J.C.P., Lankarani, H.M.: Dynamics of multibody systems with spherical clearance joints. J. Comput. Nonlinear Dyn. **1**(3), 240–247 (2006). https://doi.org/10.1115/1.2198877
29. Flores, P., Ambrósio, J., Lankarani, H.M.: Contact-impact events with friction in multibody dynamics: back to basics. Mech. Mach. Theory **184**, 105305 (2023). https://doi.org/10.1016/j.mechmachtheory.2023.105305
30. García de Jalón, J., Gutiérrez-López, M.D.: Multibody dynamics with redundant constraints and singular mass matrix: existence, uniqueness, and determination of solutions for accelerations and constraint forces. Multibody Syst. Dyn. **30**(3), 311–341 (2013). https://doi.org/10.1007/s11044-013-9358-7
31. Gonthier, Y., McPhee, J., Lange, C., Piedbœuf, J.C.: A regularized contact model with asymmetric damping and dwell-time dependent friction. Multibody Syst. Dyn. **11**(3), 209–233 (2004). https://doi.org/10.1023/B:MUBO.0000029392.21648.bc
32. Gowda, S., Ma, Y., Churavy, V., Edelman, A., Rackauckas, C.: Sparsity programming: automated sparsity-aware optimizations in differentiable programming. In: Program Transformations for ML Workshop at NeurIPS 2019 (2019). https://openreview.net/forum?id=rJlPdcY38B
33. Gowda, S., Ma, Y., Cheli, A., Gwozdz, M., Shah, V.B., Edelman, A., Rackauckas, C.: High-performance symbolic-numerics via multiple dispatch (2021). ArXiv preprint arXiv:2105.03949
34. Griewank, A., Walther, A.: Evaluating derivatives, second edn. J. Soc. Ind. Appl. Math. (2008). https://doi.org/10.1137/1.9780898717761
35. Hairer, E., Wanner, G.: Stiff differential equations solved by Radau methods. J. Comput. Appl. Math. **111**(1), 93–111 (1999). https://doi.org/10.1016/S0377-0427(99)00134-X
36. Hairer, E., Wanner, G.: Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems. Springer Series in Computational Mathematics. Springer, Berlin Heidelberg (2010)
37. Haug, E.J.: Design sensitivity analysis of dynamic systems. In: Mota Soares, C.A. (ed.) Computer Aided Optimal Design: Structural and Mechanical Systems, pp. 705–755. Springer Berlin Heidelberg, Berlin, Heidelberg (1987)
38. Haug, E.J.: Computer Aided Kinematics and Dynamics of Mechanical Systems, Volume 1: Basic Methods. Pearson College Div, Massachusetts (1989). https://doi.org/10.1016/0278-6125(92)90050-p
39. Haug, E.J.: Simulation of spatial multibody systems with friction. Mech. Based Des. Struct. Mach. **46**(3), 347–375 (2018). https://doi.org/10.1080/15397734.2017.1377086
40. Haug, E.: Computer Aided Kinematics and Dynamics of Mechanical Systems Vol II: Modern Methods. Research Gate (2021)
41. Hindmarsh, A.C.: Odepack, a systemized collection of ode solvers. Sci. Comput. (1983)
42. Hindmarsh, A.C., Brown, P.N., Grant, K.E., Lee, S.L., Serban, R., Shumaker, D.E., Woodward, C.S.: Sundials: suite of nonlinear and differential/algebraic equation solvers. ACM Trans. Math. Softw. **31**(3), 363–396 (2005). https://doi.org/10.1145/1089014.1089020
43. Holland, J.H.: Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence. MIT Press, Cambridge (1992)
44. Janusz, F., Wojtyra, M.: On the unique solvability of a direct dynamics problem for mechanisms with redundant constraints and Coulomb friction in joints. Mech. Mach. Theory **46**(3), 312–334 (2011). https://doi.org/10.1016/j.mechmachtheory.2010.11.003
45. Kelley, C.T.: Solving Nonlinear Equations with Newton's Method, Chap. 3, pp. 57–83. SIAM, Philadelphia (2003). https://doi.org/10.1137/1.9780898718898

46. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of ICNN'95 - International Conference on Neural Networks, vol. 4, pp. 1942–1948 (1995). https://doi.org/10.1109/ICNN.1995.488968
47. Kim, S., Ji, W., Deng, S., Ma, Y., Rackauckas, C.: Stiff neural ordinary differential equations. Chaos, Interdiscip. J. Nonlinear Sci. **31**(9), 093122 (2021). https://doi.org/10.1063/5.0060697
48. Knoll, D., Keyes, D.: Jacobian-free Newton–Krylov methods: a survey of approaches and applications. J. Comput. Phys. **193**(2), 357–397 (2004). https://doi.org/10.1016/j.jcp.2003.08.010
49. Koolen, T., contributors: Rigidbodydynamics.jl (2016). https://github.com/JuliaRobotics/RigidBodyDynamics.jl
50. Li, S., Petzold, L.: Software and algorithms for sensitivity analysis of large-scale differential algebraic systems. J. Comput. Appl. Math. **125**(1), 131–145 (2000)
51. López Varela, Á., Sandu, C., Sandu, A., Dopico Dopico, D.: Discrete adjoint variable method for the sensitivity analysis of ALI3-P formulations. Multibody Syst. Dyn. (2023)
52. Ma, Y., Dixit, V., Innes, M.J., Guo, X., Rackauckas, C.: A comparison of automatic differentiation and continuous sensitivity analysis for derivatives of differential equation solutions. In: 2021 IEEE High Performance Extreme Computing Conference (HPEC), pp. 1–9 (2021). https://doi.org/10.1109/HPEC49654.2021.9622796
53. Ma, Y., Gowda, S., Anantharaman, R., Laughman, C., Shah, V., Rackauckas, C.: ModelingToolkit: a composable graph transformation system for equation-based modeling (2021)
54. Marques, F., Flores, P., Claro, J.C.P., Lankarani, H.M.: Modeling and analysis of friction including rolling effects in multibody dynamics: a review. Multibody Syst. Dyn. **45**(2), 223–244 (2019). https://doi.org/10.1007/s11044-018-09640-6
55. McCourt, M., Smith, B., Zhang, H.: Sparse matrix-matrix products executed through coloring. SIAM J. Matrix Anal. Appl. **36**(1), 90–109 (2015). https://doi.org/10.1137/13093426X
56. Miranda, M.J., Fackler, P.L.: Applied Computational Economics and Finance. MIT Press Books, vol. 1. The MIT Press, Cambridge (2004). https://ideas.repec.org/b/mtp/titles/0262633094.html
57. More, J.J., Garbow, B.S., Hillstrom, K.E.: User guide for MINPACK-1. [In FORTRAN]. Web (1980). https://doi.org/10.2172/6997568
58. Muchnick, S.S.: Advanced Compiler Design and Implementation. Morgan Kaufmann Publishers, San Francisco (1997). http://catdir.loc.gov/catdir/toc/els032/97013063.html
59. Nakka, S.K.S., Alexander-Ramos, M.J.: Simultaneous combined optimal design and control formulation for aircraft hybrid-electric propulsion systems. J. Aircr. **58**(1), 53–62 (2021). https://doi.org/10.2514/1.C035678
60. Nelder, J.A., Mead, R.: A simplex method for function minimization. Comput. J. **7**(4), 308–313 (1965). https://doi.org/10.1093/comjnl/7.4.308
61. Nocedal, J., Wright, S.J.: Numerical Optimization, 2nd edn. Springer, New York (2006)
62. Orden, J.C.G.: Analysis of joint clearances in multibody systems. Multibody Syst. Dyn. **13**(4), 401–420 (2005). https://doi.org/10.1007/s11044-005-3989-2
63. Pagalday, J.M., Avello, A.: Optimization of multibody dynamics using object oriented programming and a mixed numerical-symbolic penalty formulation. Mech. Mach. Theory **32**(2), 161–174 (1997). https://www.sciencedirect.com/science/article/pii/S0094114X96000377
64. Pennestrì, E., Valentini, P.P., Vita, L.: Multibody dynamics simulation of planar linkages with Dahl friction. Multibody Syst. Dyn. **17**(4), 321–347 (2007). https://doi.org/10.1007/s11044-007-9047-5
65. Petzold, L.R.: Description of DASSL: a differential/algebraic system solver. Web (1982). https://www.osti.gov/biblio/5882821
66. Petzold, L.: Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations. SIAM J. Sci. Stat. Comput. **4**(1), 136–148 (1983). https://doi.org/10.1137/0904010
67. Petzold, L., Li, S., Cao, Y., Serban, R.: Sensitivity analysis of differential-algebraic equations and partial differential equations. Comput. Chem. Eng. **30**(10), 1553–1559 (2006)
68. Pikuliński, M., Malczyk, P.: Adjoint method for optimal control of multibody systems in the Hamiltonian setting. Mech. Mach. Theory **166**, 104473 (2021). https://doi.org/10.1016/j.mechmachtheory.2021.104473
69. Rackauckas, C.: Direct automatic differentiation of (differential equation) solvers vs analytical adjoints: Which is better? (2022). https://www.stochasticlifestyle.com/direct-automatic-differentiation-of-solvers-vs-analytical-adjoints-which-is-better/
70. Rackauckas, C., Anantharaman, R., Edelman, A., Gowda, S., Gwozdz, M., Jain, A., Laughman, C., Ma, Y., Martinuzzi, F., Pal, A., Rajput, U., Saba, E., Shah, V.B.: Composing modeling and simulation with machine learning in Julia (2021)
71. Rackauckas, C., Ma, Y., Martensen, J., Warner, C., Zubov, K., Supekar, R., Skinner, D., Ramadhan, A., Edelman, A.: Universal Differential Equations for Scientific Machine Learning (2021)
72. Radhakrishnan, K., Hindmarsh, A.C.: Description and use of LSODE, the Livermore solver for ordinary differential equations. Tech. Rep., Lawrence Livermore National Laboratory (1993)

73. Revels, J., Lubin, M., Papamarkou, T.: Forward-mode automatic differentiation in Julia (2016). arXiv:1607.07892 [cs.MS]
74. Rothwell, A.: Numerical Methods for Unconstrained Optimization pp. 83–106. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-55197-5_4
75. Serban, R., Negrut, D., Haug, E.J., Potra, F.A.: A topology-based approach for exploiting sparsity in multibody dynamics in Cartesian formulation. Mech. Struct. Mach. **25**(3), 379–396 (1997). https://doi.org/10.1080/08905459708905295
76. Steinebach, G.: Order reduction of ROW methods for DAEs and method of lines applications. Preprint. Techn. Hochsch., Fachbereich Mathematik (1995). https://books.google.com/books?id=6TRDHQAACAAJ
77. Steinebach, G.: Construction of Rosenbrock–Wanner method Rodas5P and numerical benchmarks within the Julia differential equations package. BIT Numer. Math. **63**(2), 27 (2023). https://doi.org/10.1007/s10543-023-00967-x
78. Tian, Q., Flores, P., Lankarani, H.M.: A comprehensive survey of the analytical, numerical and experimental methodologies for dynamics of multibody mechanical systems with clearance or imperfect joints. Mech. Mach. Theory **122**, 1–57 (2018). https://doi.org/10.1016/j.mechmachtheory.2017.12.002
79. Varela, A.L., Dopico, D.D., Fernández, A.L.: Augmented Lagrangian index-3 semi-recursive formulations with projections. Direct sensitivity analysis (2023). https://doi.org/10.21203/rs.3.rs-2687084/v1
80. Verulkar, A.: Direct sensitivity analysis of spatial multibody systems with joint friction. Masters Theses (2021). http://hdl.handle.net/10919/112732
81. Verulkar, A., Sandu, C., Dopico, D., Sandu, A.: Computation of direct sensitivities of spatial multibody systems with joint friction. J. Comput. Nonlinear Dyn. **17**(7), 071006 (2022). https://doi.org/10.1115/1.4054110
82. Verulkar, A., Sandu, C., Sandu, A., Dopico, D.: Optimal control of multibody systems in reduced space ODE formulation. In: IUTAM Symposium on Optimal Design and Control of Multibody Systems (2022)
83. Wächter, A., Biegler, L.: On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math. Program. **106**(1), 25–57 (2006). https://doi.org/10.1007/s10107-004-0559-y. Copyright: Copyright 2008 Elsevier B.V., All rights reserved
84. Wanner, G., Hairer, E.: Solving Ordinary Differential Equations II, vol. 375. Springer, Berlin Heidelberg New York (1996)
85. Zhu, Y., Dopico, D., Sandu, C., Sandu, A.: Dynamic response optimization of complex multibody systems in a penalty formulation using adjoint sensitivity. J. Comput. Nonlinear Dyn. **10**(3), 1–9 (2015). https://doi.org/10.1115/1.4029601