FModal: A Flexible Body Dynamics Modeling Pipeline for Guidance and Control

Carl Leake, Abhinandan Jain
Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Dr.
Pasadena, CA 91109
carl.leake@jpl.nasa.gov, abhinandan.jain@jpl.nasa.gov

Abstract—**This article describes the FModal tool that has been designed to bridge the gap between the structural dynamics and guidance and control domains to facilitate the development and use of high-fidelity flexible body dynamics models. FModal streamlines the process of generating modal data—including residual vectors and modal integral terms—from component NASTRAN structural dynamics models. The data generation process can be tailored to meet simulation fidelity and performance needs. FModal's output is a portable HDF5 file with hierarchical, well-organized, and labeled data that can be used to automate, simplify, and speed up the creation of flexible multibody dynamics models—resulting in faster design iterations and reduced costs. This paper uses an interface between FModal and the DARTS flexible multibody dynamics tool to carry out several numerical studies to exercise and validate the FModal pipeline.**

TABLE OF CONTENTS

1. INTRODUCTION

Aerospace vehicles can involve multiple coupled rigid and flexible bodies undergoing large articulation and other configuration changes. Dynamics models and system-level modes play a critical role in guidance and control (G&C) development for flexible body vehicles. The modal data for these systems is derived from finite element method (FEM) structural analysis models. The development of a FEM model is typically based on linear structural analysis and is thus limited to a specific configuration of the vehicle. G&C systems, however, are required to support a broad envelope of vehicle configuration changes including large articulation of bodies, constraint changes, attachment and detachment of bodies, etc.

Vehicle configuration changes can significantly change the system-level modes of a vehicle. As an illustration, consider a simple system consisting of two identical, flexible beams joined by a locked hinge. Figure 1 shows the system level modal frequencies when the joint angle between the beams is zero degrees and 45 degrees. Note that the system level frequencies change by a noticeable amount when the joint angle between the beams is modified. Characterizing a vehicle's system-level modal properties would seemingly require an impractically large number of FEM solutions sampling the configuration space for G&C development. Such a path corresponds to the approach shown at the bottom of Figure 2. One way to address this challenge is to utilize multibody modeling techniques for developing G&C models. Multibody dynamics modeling methods capture system nonlinearities accurately across time-varying system configurations. Reference [1] discussed the importance of capable flexible multibody dynamics tools for (G&C) development for handling:

- large articulation of bodies and small deformation of flexible bodies
- nonlinear and linearized dynamics modeling
- configuration changes from attachment and detachment of bodies
- mass property changes such as from fuel depletion

Several tool options for *rigid* multibody dynamics are available in the community. Due to rigid body limitations, body flexibility is often mimicked using methods such as substructuring, i.e., approximating a flexible body via a set of rigid bodies connected by springs and dampers. This is the middle path shown in Figure 2. Frequent and tedious tuning of the stiffness and damping coefficients is required to match system-level modal responses. In the earlier two-beam example, changing the beam angle from zero degrees to 45 degrees would require retuning the springs and dampers to match the system modal response. This rigid multibody tool-based modeling approach is also fragile and expensive.

This paper proposes a *flexible* multibody dynamics based approach, which overcomes many of the earlier challenges, and is depicted in the upper path in Figure 2. This approach captures configuration-dependent, system-level modal property changes accurately when there are a small number of FEM models. When applied to the two-beam problem, the system-level modal frequency changes shown in Figure 1 were reproduced automatically to within 10^{-7} Hz using just a single FEM model and without requiring tuning such as with rigid multibody models. From a G&C standpoint, this approach enables the design and evaluation of G&C performance over a large vehicle configuration space with minimal need for multiple FEM models. The general impediment to pursuing this approach has been the very limited availability

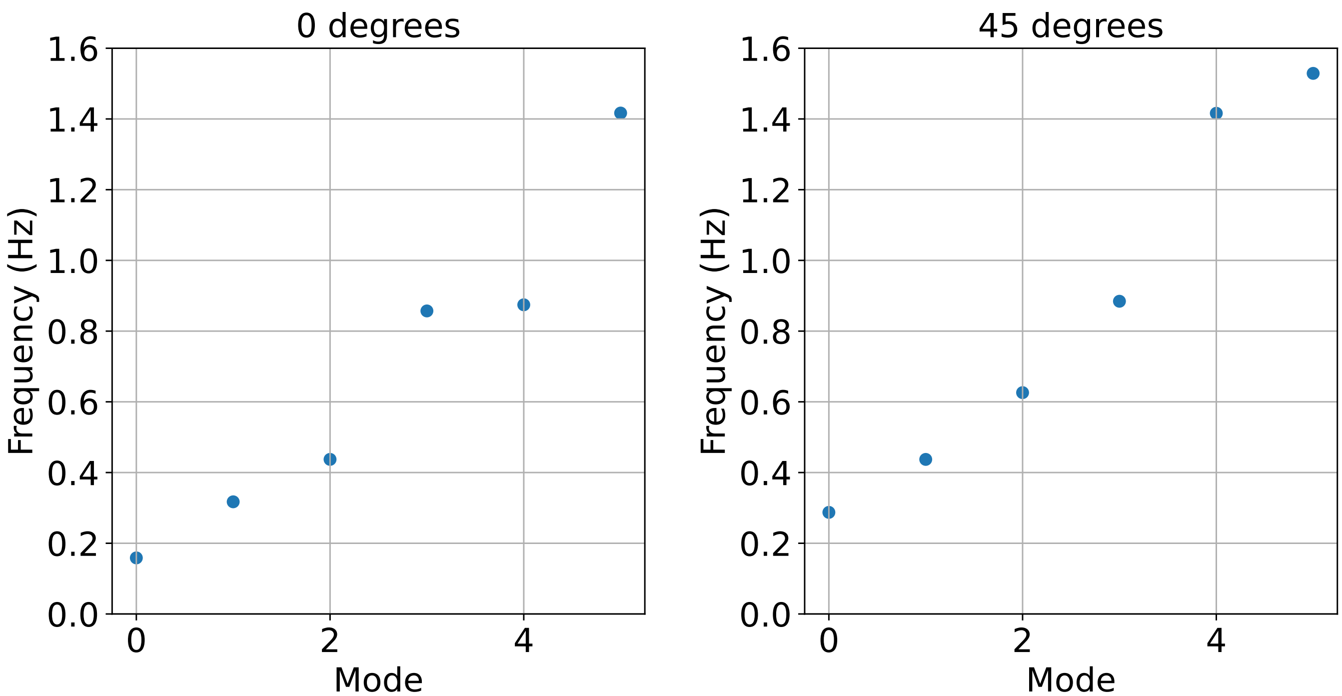


Figure 1: Modal frequencies of two locked beams joined at zero degrees (left) and 45 degrees (right).

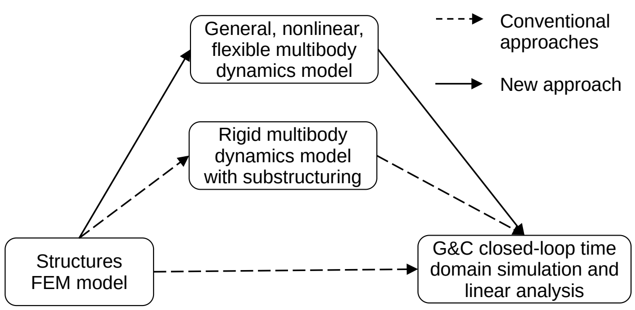


Figure 2: Capable flexible multibody tools provide a path to high-fidelity G&C dynamics models

of general purpose flexible multibody dynamics tools.

Even with the availability of a flexible multibody dynamics tool, processing and transferring data from a FEM model to the multibody dynamics tool is a complex process that involves multiple steps, such as:

1. structural dynamics FEM-based modal analysis for each flexible body
2. computation and transfer of a large set of quantities such as frequencies, mode shapes, modal integrals, etc. from modal analysis
3. use of this data to properly create and connect corresponding flexible bodies within the multibody dynamics model

The modal data generation process requires proficiency with both FEM and multibody software as well as the careful and error-free generation and transfer of a large amount of data across tools. The process can be quite challenging, slow, and fragile for even simple flexible body models. An unsatisfactory consequence of these difficulties has seen the G&C community resorting to simplified and low-fidelity vehicle dynamics models.

One of the key goals of the **FModal** tool described in this article is to avoid the unnecessary devaluation of flexible body dynamics model fidelity. FModal provides a pipeline to bridge the gap between the structural dynamics and G&C domains. It streamlines the process of generating modal data from component NASTRAN structural dynamics models for use in flexible multibody dynamics tools. It provides a simple Python interface that allows users to easily tailor the fidelity and performance of the model. FModal's output is a portable HDF5 file containing a hierarchical, well-organized, and labeled data set that can be read by multibody tools to obtain the body data. The FModal tool can thus be used by G&C analysts to extract the critical data needed for flexible multibody dynamics simulations from NASTRAN structures models. This can speed up the process of creating flexible body models leading to more capable and faster design iterations and reduced costs.

The remainder of the article is organized as follows. Section 2 contains a description of flexible body dynamics models. Section 3 provides an overview of the FModal pipeline. Section 4 discusses G&C modeling workflows that use the FModal capability. Section 5 contains a discussion of various numerical studies that have been carried out using JPL's DARTS flexible multibody dynamics tool to validate the FModal pipeline. The reader should refer to Ref. [1] for additional details and background material beyond what is covered in this article.

2. Flexible Body Dynamics Models

This section briefly describes the approach for modeling the dynamics of flexible multibody systems.

Nonlinear Multibody Dynamics

Multibody dynamics methods are a well-studied topic within the technical community. These methods are commonly used to model the dynamics of a wide range of engineering systems, including aerospace vehicles, ground vehicles, and robotics platforms. A schematic of a generic multibody is shown in Figure 3. The dynamics models of multibody systems are nonlinear, since the motions of the component rigid and flexible are fully coupled. The primary approaches for modeling multibody systems include *minimal-coordinate* methods [2], [3] and *constraint-based* methods [4], [5], [6]. The minimal coordinate methods are more complex but provide recursive algorithms that are low-cost and better behaved. Constraint-based methods, on the other hand, are simpler to implement, but are computationally inefficient and require more complex differential algebraic solvers to manage constraint errors. Due to its advantages, the minimal coordinate approach is pursued here.

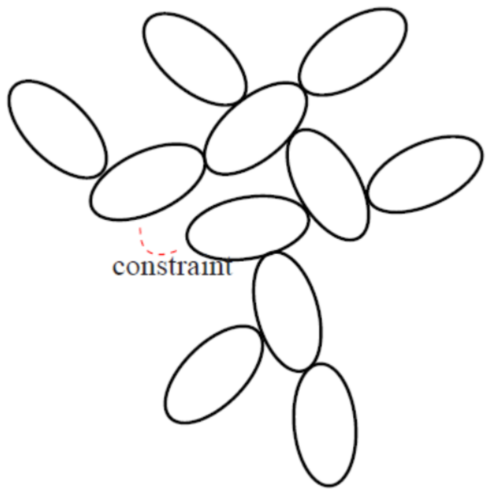


Figure 3: Tree-structured multibody system with cut-joint closure constraints

The minimal coordinate equations of motion (EOM) of a tree-topology flexible multibody system have the form [2]:

$$\mathcal{T} = \mathcal{M}(\boldsymbol{\vartheta})\ddot{\boldsymbol{\vartheta}} + \mathcal{C}(\boldsymbol{\vartheta}, \dot{\boldsymbol{\vartheta}}) \tag{1}$$

where $\mathcal{M}(\boldsymbol{\vartheta})$ denotes the mass matrix and $\mathcal{C}(\boldsymbol{\vartheta}, \dot{\boldsymbol{\vartheta}})$ the gyroscopic and Coriolis terms. The $\boldsymbol{\vartheta}$ generalized coordinates contain the hinge articulation coordinates as well as deformation modal coordinate DoFs for all the bodies in the system. The rigid body and elastic DoFs are fully coupled. Equation 1 provides an exact model valid for large hinge articulation and small elastic deformations of flexible bodies [1]. This model also takes into account the deformation-dependent variation of flexible body mass properties. So-called *modal integrals* provide a cost-effective way to compute these variable effects.

For flexible multibody systems with closed-chain topology, cut-joints are used to decompose the multibody model into a tree-topology system (such as in Eq. 1) together with additional inter-body, cut-joint closure constraints. A simple example of such a decomposition is a fixed-fixed beam modeled with one end attached via a locked joint and the other end fixed via a closure constraint.

Flexible Body Data from FModal

Flexible body data, e.g., mode shapes, frequencies, and modal integrals, for individual bodies required by the multibody model can be derived offline from the corresponding FEM model data for each body. *FModal* is a C++ and Python toolkit designed to process NASTRAN FEM bulk data and compute and extract said flexible body data for use in a multibody dynamics model. The following subsections describe the specific flexible body data generated by FModal for a single-component flexible body.

Mode Shapes

FModal extracts normalized component modes from a body's FEM data. An outline of the process is as follows (see Ref. [1] for more details):

1. Using the nodal DoFs in the *a-set*[2] as the boundary (interface) DoFs, calculate the Craig-Bampton transformation [7]; the a-set nodal DoFs can be supplied by the user or automatically generated by the FModal software.
2. Using the Craig-Bampton system, solve for the mode shapes using the standard eigenvalue/eigenvector problem and normalize using the mass matrix.

These mode shapes diagonalize the stiffness matrix. The rigid body modes can be easily removed using a frequency cutoff and the remaining modes can be used to constrain the boundary (interface) nodes. By default, these are the mode shapes used by FModal. One can obtain *free-free* modes from FModal by not including any nodes in the a-set.

Modal Integrals

Modal integrals are higher-order terms that appear in the system's kinetic energy and couple the rigid and flexible body dynamics. In addition, they account for the deformation-dependent variation in a body's moments of inertia and the center of mass location. Their effects can be significant for systems that experience large angular rates and accelerations. Modal integrals can be calculated using the mass matrix and mode shapes; the details can be found in Appendix E of Ref. [1]. FModal supports the computation of these often-neglected modal integrals, which facilitates their inclusion within dynamics models with little effort.

Residual Vectors

Reduced order dynamics models for G&C development are often developed by truncating the set of normalized component modes based on a cut-off frequency. Beyond the impact on the dynamics, such truncation may cause noticeable errors in the static deformations of flexible bodies as well. Accurate static deformations can be important for the generation of linearized state-space models that are often used in G&C analyses since linearizations are typically done about static equilibrium. The situation is similar for modal analyses.

Residual vectors (also called modal truncation vectors in the literature) can be used to recover the static behavior at the cost of adding DoFs to the reduced-system model. These additional DoFs, however, are typically far fewer than the number of DoFs removed by the truncation process. Appendix A provides an overview of the concept of residual vectors. FModal supports the computation of residual vectors since they can significantly improve the accuracy of determining

[2]In NASTRAN, the a-set is a set of nodal DoFs that the user intends to constrain. For example, a cantilever beam that is cantilevered at node 1 would have all six DoFs of node 1 in the a-set.

equilibrium states for linearized models. The residual vectors are computed for a user-specified set of nodal DoFs. Often, these are the nodal DoFs where external forces are applied by attached actuators to the flexible body. FModal uses this information to modify the NASTRAN run deck to enable the calculation of residual vectors.

3. FMODAL DESIGN

This section describes key details about the design of the FModal tool.

FModal Pipeline

Figure 4 illustrates a typical FModal workflow [1]. FModal imports and reads NASTRAN FEM models by parsing the executive, case control, and bulk data sections of a NASTRAN run deck for a component body. This import can consist of complex NASTRAN run decks with many "include" statements. After importing the run deck, FModal creates a custom NASTRAN run deck used to generate the flexible body data needed for G&C flexible body dynamics models. This includes normalized component mode shapes and frequencies (see section 2), grid point locations, pose of all coordinate systems, modal integrals (see section 2), rigid body mass properties, coupled damping matrices, Craig-Bampton modes, child bodies (lumped masses that were removed and are now standalone rigid bodies), and the global physical model in sparse matrix format. FModal output data is stored in a portable HDF5 file format [8]. The HDF5 format is well supported by standard software languages (e.g., Matlab, Python, Mathematica), is self-documenting, and suitable for large data sets. An example of the groups that make up such an HDF5 file is shown in Appendix D.

Each flexible body's HDF5 file generated by FModal contains the data needed to include the body in a multibody dynamics model. The repeated use of FModal for each of the flexible bodies streamlines the process for developing full non-linear flexible multibody dynamics models.

FModal Software Classes

The FModal tool implements a `Nastran` Python class to automate the creation of flexible body data from a NASTRAN model. The `Nastran` class takes a user-specified NASTRAN model, performs modal analysis based on user-specified options (described in section 2), and saves all critical data to an HDF5 file. This process is facilitated in part via NASTRAN DMAPs. Users can tailor the output data, e.g., the inclusion of modal integrals, via keyword arguments to the `Nastran.run_103`[3] or `Nastran.load_op2`[4] functions. `Nastran.run_103` runs the NASTRAN 103 solution with the user-specified options, and `Nastran.load_op2` post-processes the data from the 103 solution and packages it in an HDF5 file. All necessary modifications to the NASTRAN model and/or post-processing are taken care of internally by the `Nastran` class. The key options for the `Nastran.run_103` function are described below:

- `n_modes` - Specifies the number of flexible modes calculated and stored during modal analysis.
- `csid` - Specifies the CSID used by NASTRAN; this is the output coordinate system ID.
- `aset` - A dictionary that specifies which nodal DoFs should be added to the a-set.
- `rvdof` - A dictionary that specifies which nodal DoFs should be used in the residual vector calculations.
- `grdpnt` - Specifies the ground point to be used.
- `separateLumpedRigidBodies` - A boolean that specifies whether lumped masses should be separated from the NASTRAN model and processed separately. The information for these lumped masses is stored in a separate section of the output HDF5 file and can be used to create and attach them as rigid bodies in the multibody dynamics model.
- `addLumpedRigidBodiesToAset` - A boolean that specifies whether the constrained DoFs of the separated lumped rigid bodies' nodes should be added to the a-set.

The key options for the `Nastran.load_op2` function are described below:

- `fop2` - OP2[5] file name. If none is specified, a name based on the name of the NASTRAN model is used.
- `grdpnt` - The ground point to use for calculations.
- `modal_integral_flag` - Boolean that determines whether modal integrals are calculated or not.
- `rigid_body_mode_thresh` - The frequency threshold used to determine if a given modal frequency is a rigid body mode or a flexible body mode.

These functions can be used to easily create flexible multibody dynamics models and tailor their fidelity to meet the user's simulation needs by simply modifying a few Python arguments. Little expert knowledge of NASTRAN is required. An example of a `Nastran` run script is included in Appendix B.

Handling Common Configuration Changes with the Multibody Model

Configuration changes to a vehicle from body articulation, separation events, mass property changes, etc. affect the system-level modal properties. Understanding their impact on the dynamics of vehicles is important for G&C systems since they are expected to perform across a broad range of such vehicle configuration changes. The conventional approach for handling configuration changes is to generate a family of FEM data for a representative sampling of the anticipated configurations within the design envelope. Each such FEM data set generation requires expensive and time-consuming structural analysis, FEM data generation, and modal analysis solutions.

FModal provides a simpler way to generate such a family of dynamics models across the configuration space. In this new approach, FModal is used to separately generate the modal data for each rigid/flexible body involved in configuration changes. These modal data sets for the component bodies are assembled into a multibody model, and configuration changes are handled by changes within the multibody model. The number of system-level structural analysis FEM models required is significantly reduced in this approach.

The most common case of a configuration change arises from large angle articulation of bodies, such as solar panels, scan platforms, and engine gimbals. Such changes are easily accommodated within multibody models via the articulation of joints without requiring additional FEM models.

[3]The 103 solution is used by NASTRAN for modal analyses.

[4] An OP2 file is a binary NASTRAN data file that is output when running FModal.

[5]See footnote 4.

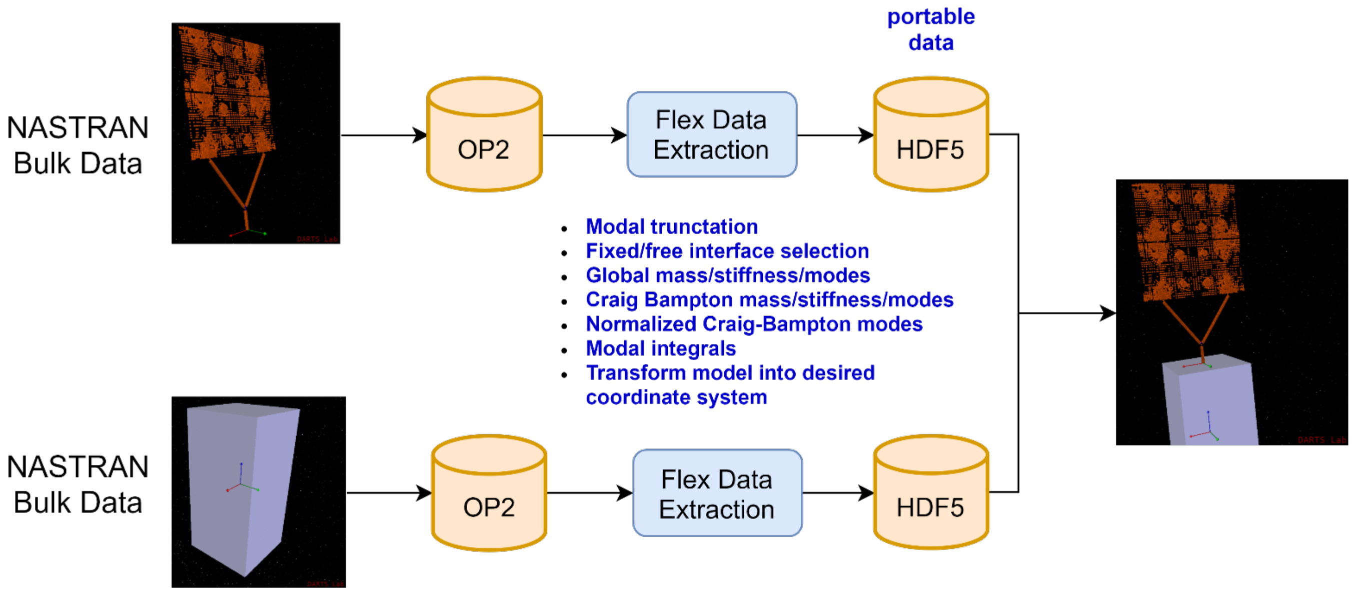


Figure 4: Component synthesis of spacecraft bus and solar array models using FModal

Handling Lumped Masses in a Flexible Body Model

Oftentimes, FEM models include lumped masses that undergo changes or separation events over time, e.g., solid rocket boosters (SRBs) on a launch vehicle. The FEM for the vehicle's core stage may have the SRBs represented as lumped masses. To avoid the regeneration of FEM models due to changes in these lumped masses, FModal provides an option to separate said lumped masses from the rest of the flexible body when generating the modal data. The multibody model can then recreate the system by creating the flexible body and attaching to it rigid bodies for each of the lumped masses. Subsequent changes to these rigid bodies are easily handled within the multibody setting.

As mentioned in the description of the FModal classes, the `separateLumpedRigidBodies` and `addLumpedRigidBodiesToAset` keywords can be used to separate out the lumped masses from the flexible body. In the launch vehicle example, the SRB lumped masses would appear as separate, rigid bodies attached to the core stage flexible body in the multibody dynamics model. It is easy to change the mass of the SRBs as fuel burns and to detach them during a separation event within the multibody model; this avoids the need for new FEM models.

Currently, this procedure only handles CONM2 lumped masses. It can handle lumped masses connected to a grid point of the flexible body, or a lumped mass connected to a grid point of the flexible body via an RBAR element. If `separateLumpedRigidBodies` is enabled, then all CONM2 entries are removed from the bulk data file, and their mass properties and connection information—joint type, connection point, etc.—are stored in the resultant HDF5 file in the ChildBodies group. In addition, if `addLumpedRigidBodiesToAset` is enabled, then the associated DoFs of the grid point on the flexible body are added to the a-set. This ensures that the mode shapes of the full flexible-body-plus-rigid-child-bodies system will be correct when reconnected in the multibody dynamics simulation.

4. FModal Modeling Workflow

This section describes notional workflows for FModal and multibody-modeling-based flexible body dynamics development for G&C. The goal of the workflow is to generate, using a small amount of FEM data, high-fidelity flexible multibody dynamics models that can cover a broad range, ideally the entire range, of the vehicle's configurations.

A key part of the workflow is the dynamics model development. This will typically involve the following steps:

- Identify a reference vehicle configuration.
  - Get system-level FEM data for the overall system.
  - Get FEM data for component bodies. If familiar with NASTRAN, these can be broken out from the system FEM locally.
  - For each flexible body:
    - Extract lumped masses.
    - Define a-set DoFs for hinges and closure constraints.
    - Define residual vector DoFs, i.e., nodal DoFs where forces will be applied.
    - Define the number of modes.
    - Run FModal to extract the flexible body data and save it into an HDF5 file.
- (Optional) Identify additional vehicle configurations that take into account articulation, topology, closure constraints, force application, mass properties, etc. For each of these, request system-level FEM data. This data will primarily be used for validation of the multibody model and occasionally for developing new multibody models for configurations far from the main one.
- Assemble the multibody:
  - Create flexible bodies with attached rigid bodies for the lumped masses using the HDF5 file data.
  - Connect the bodies via hinges.
  - Create nodes where forces will be applied and sensing will be done.
- Initial validation:
  - Set up system coordinates to match the initial system.
  - Linearize the multibody model and extract a state-space model.
  - Extract modes.

– Compare with system modes and verify that they match those from the system FEM model.

The previous steps are used to create a flexible body multibody dynamics model. Now, let's focus on its usage. For G&C, the usage can vary across the analysis, design, verification, and validation stages. Usually, the focus is on the closed-loop system. This can require:

• Developing and integrating additional models for actuators, sensors, and the environment (e.g., aerodynamics, gravity, etc.) into the multibody model to develop a system-level dynamics model. Such models are needed for studying servo-elastic effects.
• Closing the loop around the control algorithms and software to develop an overall closed-loop dynamics model. This is the overall nonlinear system model whose design and performance need to meet G&C requirements

Depending on the G&C development and usage phase, there are multiple ways that this dynamics model may be used:

• Carry out time-domain simulations with the high-fidelity, non-linear, flexible multibody model to evaluate overall system performance. Note that within this multibody model, it is possible to:
 – Change coordinates to articulate bodies
 – Apply forces
 – Change mass properties
 – Attach/detach bodies and change closure constraints
• Generate linearized system models that can be used to generate Bode plots etc. for linear G&C analysis. Note that these models are limited to the narrow configurations around the linearization state.

Changes in vehicle configuration will require modifications at one or more levels; in order of increasing complexity to the developer: (1) changes made within the multibody dynamics model, (2) changes that require rerunning FModal and updating the multibody model, and (3) changes that require changing the starting FEM model and the steps that follow. Changes such as modifying mass properties, applying different force models, or changing joint angles only require changes to the multibody dynamics model.

Larger changes such as attaching/detaching bodies may require a new model via FModal. For example, changing the joint that attaches two bodies from a locked joint to a pinned joint will require rerunning FModal on the original FEM, but with different nodal DoFs in the a-set. To allow handling such configuration changes on the fly, the user can run both of these cases through FModal offline, and then change the flexible body data—mode shapes, etc.—at runtime within the multibody dynamics simulation when the joint type changes. While there may be cases that at first appear to require multiple passes through FModal, this may not always be necessary. One example is a flexible body with an applied external force that changes its point of application during the simulation. At first, it may seem that two models are needed from FModal: one with residual vectors for the DoFs associated with the first point of application and another with residual vectors for the DoFs associated with the second point of application. However, a better solution for this case is to create a single model via FModal that has residual vectors that encompass both points.

More drastic changes to the underlying flexible body may require changing the FEM model. For example, if the fuel fill of the core stage of a launch vehicle significantly affects the stiffness of the core stage, then FEM files for the core stage at different fuel fills may be needed to accurately simulate the launch vehicle.

5. Numerical Results

This section describes numerical studies and cross-validation tests that have been carried out to exercise and validate the FModal pipeline. While the FModal-generated HDF5 files are portable and can be used with any flexible multibody dynamics tool, in this paper, JPL's DARTS flexible multibody dynamics tool is used together with FModal to carry out these numerical studies. The following subsection provides an overview of DARTS.

DARTS Overview

JPL's DARTS is a general-purpose software for flexible multibody dynamics modeling, analysis, and simulation [9]. DARTS uses minimal coordinate models based on the *Spatial Operator Algebra (SOA)* methodology [2], which provides low-cost recursive computational algorithms for solving the EOMs.

DARTS is designed to handle rigid/flexible multibody dynamics, arbitrary system topologies, smooth and non-smooth dynamics, and run-time configuration changes. In addition, it provides a full complement of algorithms for dynamics analysis and model-based control with fast computational performance. While the DARTS object-oriented implementation is in C++, a rich Python interface is available for all classes and methods in the system. This allows users full flexibility in defining and configuring the model as desired and modifying the model topology and properties during runtime. DARTS is used for dynamics simulations of aerospace vehicles, ground vehicles, robotics, and multi-scale molecular dynamics applications [10].

DARTS computational algorithms are structure-based and utilize scatter/gather recursions that proceed across the bodies in the system topology. This allows DARTS to be a general-purpose tool that requires no change to the software to model multibody systems with arbitrary numbers of bodies and branching structure. This property also allows DARTS to easily handle run-time structural changes in the system topology, such as attachment/detachment and addition/removal of bodies. Such structural changes are common in aerospace separation and deployment scenarios and during robotics manipulation. The algorithms accommodate such changes with recursions simply following the new system topology.

The SOA method for the dynamics of general graph topology flexible body systems is to use cut-joints to decompose the system into a flexible tree topology multibody system together with a set of inter-body closure constraints as shown in Figure 3. This method is described in chapter 11 of Ref. [2] and is available in DARTS. This procedure also generalizes the operational spatial inertia calculations to the articulated body inertia quantities for flexible bodies. Details on the validation of closed-chain dynamics for flexible multibody systems can be found later in this paper.

`ReadHDF5Flex` is a Python class in the FModal module that helps the user automate the creation of a DARTS flexible body model from the HDF5 file generated by the `Nastran` class. A similar class can be implemented to automate this process for any multibody dynamics simulation capable of simulating flexible bodies. `ReadHDF5Flex` reads the information from the HDF5 file and constructs a Python dictionary which can be used to create a flexible body within DARTS. Similar to the `Nastran` class, keywords can be used to customize the output, e.g., the mass/distance units can be transformed using user-defined parameters.

Choice of Modes

Bodies, including flexible ones, are connected via articulable hinges within a multibody model. The proper use of normalized-component modes, instead of plain free-free modes, is especially important for such inter-connected component bodies. The significance of this choice is demonstrated here by considering the spacecraft bus and solar array models shown in Figure 4. The bus model is assumed rigid, while the solar array is a flexible body with component NASTRAN FEM data.

A DARTS multibody model was constructed where the solar array FEM data was processed using the FModal HDF5 output file and attached as a flexible body to a rigid bus body. Additionally, a NASTRAN system model was constructed where the solar array component FEM model is attached to the rigid-body bus model. Finite differencing was used to generate a linear dynamics model from the nonlinear DARTS model and used to compute system modes. A figure of merit for the DARTS model is how well it reproduces the NASTRAN system-level mode shapes and frequencies. Figure 5 shows system frequency comparisons of the DARTS and NASTRAN linear models using free-free and normalized component mode shapes for the solar array body. While the differences are significant for the free-free modes case, with normalized component modes, the difference is on the order of 10^{-5} Hz. This comparison clearly illustrates the importance of using normalized component modes.

Furthermore, within the DARTS model, the G&C analyst can articulate the solar array and regenerate modal data without running the full NASTRAN system model for the new angle. For more information on this comparison, see Appendix C of Ref. [1].

Validation Approach

A number of the numerical studies described in this section compare and cross-validate the FModal/DARTS-based multibody dynamics model results with those obtained from *alternative* dynamics models of the system. Unless noted otherwise, all of the cross-comparisons showed good agreement. The specific types of comparisons carried out are listed below:

- **Derivative comparison:** For a pair of models, randomly generate an initial state (eg. position and velocity values) and apply it to both systems, carry out derivative (e.g., acceleration) computations, and cross-compare the resulting derivative values. This is repeated multiple times with different initial states each time.
- **Time-domain comparisons:** For a pair of models, run equivalent time-domain simulations. Then, compare the final values of interest between the models. For example, given a cantilevered beam attached to a rigid body with a pin joint, spin the rigid body about the pin with a prescribed acceleration for 20 seconds and cross-compare the endpoint deflections of the free ends of the beams.
- **Round-trip comparisons:** For a given model, compute the forward dynamics, followed by the inverse dynamics. Then, verify that the forces computed by the inverse dynamics match the input forces used for the forward dynamics.
- **Modal comparisons:** Compute the system-level modal frequencies and mode shapes of the DARTS model and compare them with those from an alternative (usually FEM) model.
- **Static response:** For a pair of DARTS and NASTRAN models, apply a load and solve for the static responses. In DARTS, this is done by applying the load and allowing the system to settle, whereas in NASTRAN this is done using SOL 101.

Single Constrained/Unconstrained Body

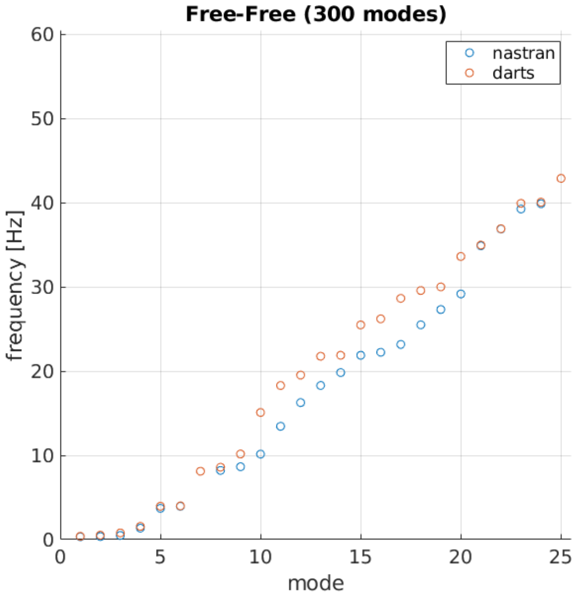
The first and simplest validation tests carried out were for a single beam. The cross-validation tests included:

- *Derivative comparisons* between a DARTS model and a *hand-derived analytical* model for a *free-free beam* and a *cantilevered beam*. These checks were done with and without the addition of modal integrals. These tests verified that the results of the EOMs within DARTS matched those of the hand-derived EOMs. These comparisons were done for a single beam for which the EOMs were straightforward to derive by hand.
- *Derivative comparisons* between a DARTS model and a *symbolic* EOM model for a *free-free beam*, a *cantilevered beam*, and a *pinned-free beam*. The comparisons were again done with and without the addition of modal integrals.
- *Round-trip comparisons* for a *free-free beam* in DARTS with modal integrals and for a *pinned-free beam* in DARTS with modal integrals. This test, which includes round trips of forward and inverse dynamics, builds on the analytical and symbolic EOM cross-validation tests described above.
- *Time-domain comparison* between a DARTS model and an *analytical model* for a *spinning free-free beam*. These were done with and without the addition of modal integrals. In contrast with the previous validation tests which were done for a single instant in time, this test simulates the dynamics over a time interval and compares the results with an analytical model.
- *Time-domain comparisons* between a DARTS model and an *analytical model* for a *cantilevered beam attached to a rigid body* spinning about a pin with a prescribed acceleration. The comparisons were done with and without the addition of modal integrals. This test is similar to the previous one, but uses a model with a different joint for the flexible body and includes an additional rigid body. This cross-validates the DARTS model for a system that includes rigid and flexible bodies. The comparison results from this case with modal integrals enabled are shown in Figure 6. The deflection vs. time of the analytical beam and DARTS beam is shown on the left, and the difference between the two systems' deflections is shown on the right. The right-hand plot shows that the difference in deflections is on the order of 10^{-10}. This good agreement between the two systems shows the FModal pipeline creates the correct model for a cantilevered beam attached to another body subject to prescribed acceleration.

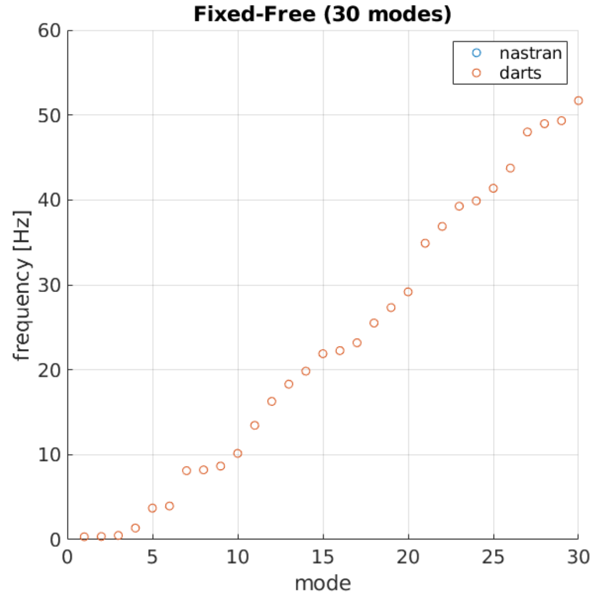
These tests also helped verify that the computation and transfer of modal integrals from NASTRAN to DARTS are being done correctly.

System Modes with Multiple Bodies

While the tests in the previous section were largely for single flexible bodies, the tests described in this section involve multiple flexible bodies—each with its own set of modal data. The resultant system's modal properties—modal frequencies and mode shapes—were compared between DARTS and

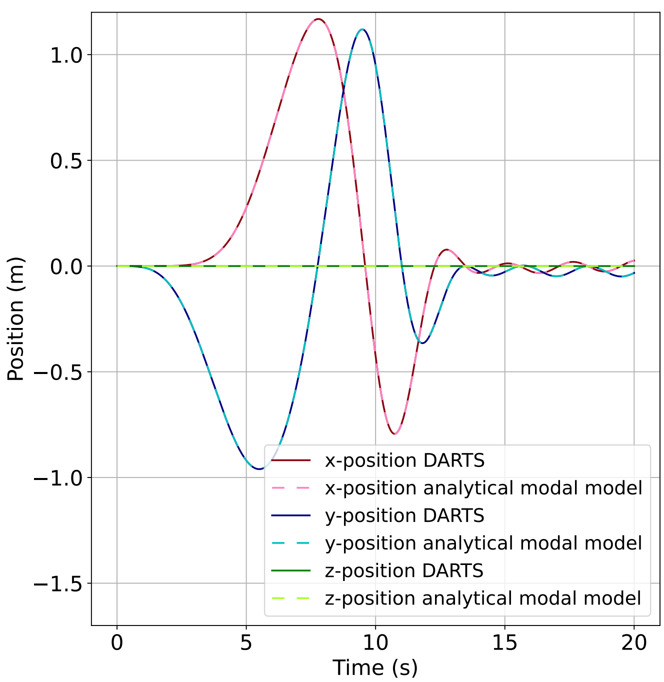


(a) Solar array processed with free-free mode shapes



(b) Solar array processed with normalized component mode shapes: fixed-free in this scenario

Figure 5: System frequency comparison between DARTS and NASTRAN using free-free and fixed-free mode shapes



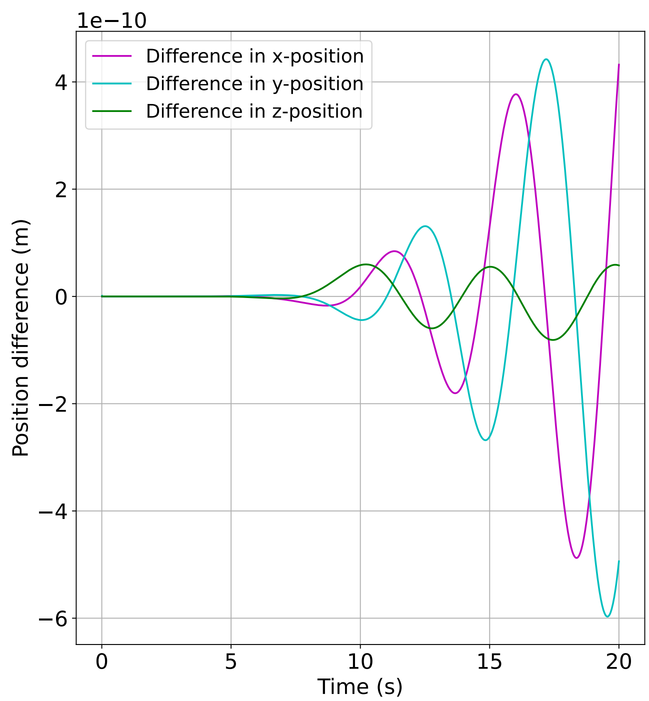


Figure 6: (Left) Deflection of the analytical beam and DARTS beam (Right) Difference in deflections between the analytical and DARTS models

alternative models.

• *Modal comparison* between a *double-length free-free beam* model in NASTRAN and a model of two *single-length beams* locked together in DARTS. The two single beams locked together represent a two-body equivalent system to the double-length beam. This provides a basic test that shows DARTS is able to correctly join multiple flexible bodies.

• *Modal comparison* between a *double-length beam* model where the two beams are joined together at a 45-degree angle in NASTRAN and an equivalent model in DARTS using two *single-length beams* locked together at a 45-degree angle. This test is similar to the previous one, except the system is connected differently. Note, that this difference changes the system frequencies noticeably: the first frequency is nearly doubled (see Figure 1). The two single beams used in the DARTS model are the same as in the previous validation script, i.e., the models produced by FModal are the same in both cases, but are joined at different angles. This provides a basic test that the multibody system can undergo large articulation at the joints and still correctly recover the system's flexible body modes.

• *Modal comparison* between DARTS and NASTRAN for *two beams pinned* together. The purpose of this test is similar to the first, except that the model complexity is increased by using a joint between the flexible bodies that allows for large angle articulation.

• *Modal comparison* between a *double-length cantilever beam* in NASTRAN and two *single-length beams* in a DARTS model, where one end of a single-length beam is fixed to a wall and its other end to the second single-length beam. This test has a similar goal to the previous two but with increased complexity from adding an extra joint. Figure 7 shows the first six modes for NASTRAN (left) and DARTS (right). Qualitatively, the mode shapes in DARTS look the same as the mode shapes in NASTRAN. Quantitatively, the Frobenius norm of the difference between the mode shape matrix in DARTS vs. NASTRAN is less than 6×10^{-6}, and the absolute difference in the modal frequencies is less than 5×10^{-6}.

A conclusion from the above cross-validation results is that the FModal/DARTS model can be used to accurately extract system mode shapes and frequencies using component body models for different rigid-body joint angle configurations. Thus, for example, one can extract the mode shapes and frequencies of a rocket with the engines at different gimbal angles via a simple for-loop in Python. This FModal/DARTS process is much simpler compared with an alternative process based on NASTRAN alone. The latter process requires: (1) modifying the NASTRAN bulk data file for each of the gimbal angle configurations of interest; (2) running a NASTRAN modal analysis for each of these cases; and (3) extracting the mode shapes and frequencies for each case.

Configuration Changes

The goal of the cross-validation tests so far has been to verify that DARTS correctly assembles a multibody model from component flexible dynamics models. The cross-validation test described below verifies that changes to mass properties and separation events are handled correctly by DARTS as well.

This test consisted of a *modal comparison* between FModal/DARTS and NASTRAN for a *free-free beam* with a detachable lumped mass. The DARTS multibody model was composed of a flexible body with a child rigid body for the lumped mass. Figure 8 shows the first six modes for this validation test for NASTRAN (left) and DARTS (right). Qualitatively, the mode shapes in DARTS look the same as the mode shapes in NASTRAN. Quantitatively, the Frobenius norm of the difference between the mode shape matrix in DARTS vs. NASTRAN is less than 5×10^{-6}, and the absolute difference in the modal frequencies is less than 5×10^{-6}.

Furthermore, this verifies that the deformable body will have the correct mode shapes before and after configuration changes arising from mass modifications or separation events. The ability to handle such configurations within an FModal/DARTS-based multibody model is much simpler than the multiple NASTRAN runs that would be required otherwise.

Closure Constraints

These tests validate that closed-chain system dynamics involving flexible bodies are being handled correctly by DARTS. Cut-joints are used to decompose the system into a tree topology system together with closure constraints during the overall solution process. Thus, a fixed-fixed beam closed-chain system is modeled as a fixed-free beam tree topology system with a cut-joint closure constraint at one of the ends.

• *Modal comparison* between DARTS and NASTRAN for a *fixed-fixed beam*. As mentioned above, a fixed-fixed beam requires a cut-joint closure constraint at one of the ends in order to obtain a tree-topology system. This test verifies that DARTS can correctly model the mode shapes and frequencies of a fixed-fixed beam. Figure 9 shows the first six modes for the validation test involving the modal comparison of a fixed-fixed beam: NASTRAN (left) and DARTS (right). Qualitatively, the mode shapes in DARTS look the same as the mode shapes in NASTRAN. Quantitatively, the Frobenius norm of the difference between the mode shape matrix in DARTS vs. NASTRAN is less than 2×10^{-6} and the absolute difference in the modal frequencies is less than 1×10^{-8}.

• *Derivative comparisons* between two DARTS systems. The system models used here were (1) a *cantilevered beam* created using a locked hinge and (2) a *cantilevered beam* created by adding a closure constraint to a free-free beam. An earlier test verified that the DARTS cantilevered beam EOMs were correct without closure constraints. This test verifies that the equivalent model created using closure constraints produces the same results.

• *Derivative comparisons* between two DARTS systems: (1) a *cantilevered beam* with a ball joint (2) a *free-free beam* with a ball joint closure constraint. This test is similar to the previous one but differs in the use of a different hinge/closure constraint type. This test further validates that EOMs using flexible body closure constraints in DARTS are correct.

• *Derivative comparisons* between two DARTS systems: (1) a *double-length cantilevered beam* created using locked joints (2) a *double-length cantilevered beam* created by adding locked closure constraints to two *free-free single-length cantilever beams*. This test is similar to the previous two, but uses different hinge/closure constraint types and adds a flexible body. This adds more complexity and further verifies the EOMs using flexible body closure constraints in DARTS.

Validating the flexible body closure constraints means that one can constrain flexible bodies any way they like, i.e., they are not limited to tree topology systems.

Residual Vectors

The tests in this section validate that residual vectors are successfully computed in NASTRAN and transferred to DARTS via the FModal pipeline.

• *Modal comparison* between NASTRAN and DARTS for a single *free-free body*. This validates that data is being transferred properly from NASTRAN to DARTS. This test validates the first step in adding residual vectors to DARTS: successfully calculating the information in NASTRAN and transferring the data correctly.

• *Static response comparison* for a *cantilevered beam* with a force applied at the free end. This comparison is done using (1) the full system in NASTRAN, (2) the DARTS system with 17 normal modes, and (3) the DARTS system with 10 normal modes plus 7 residual vectors. The validation is done by

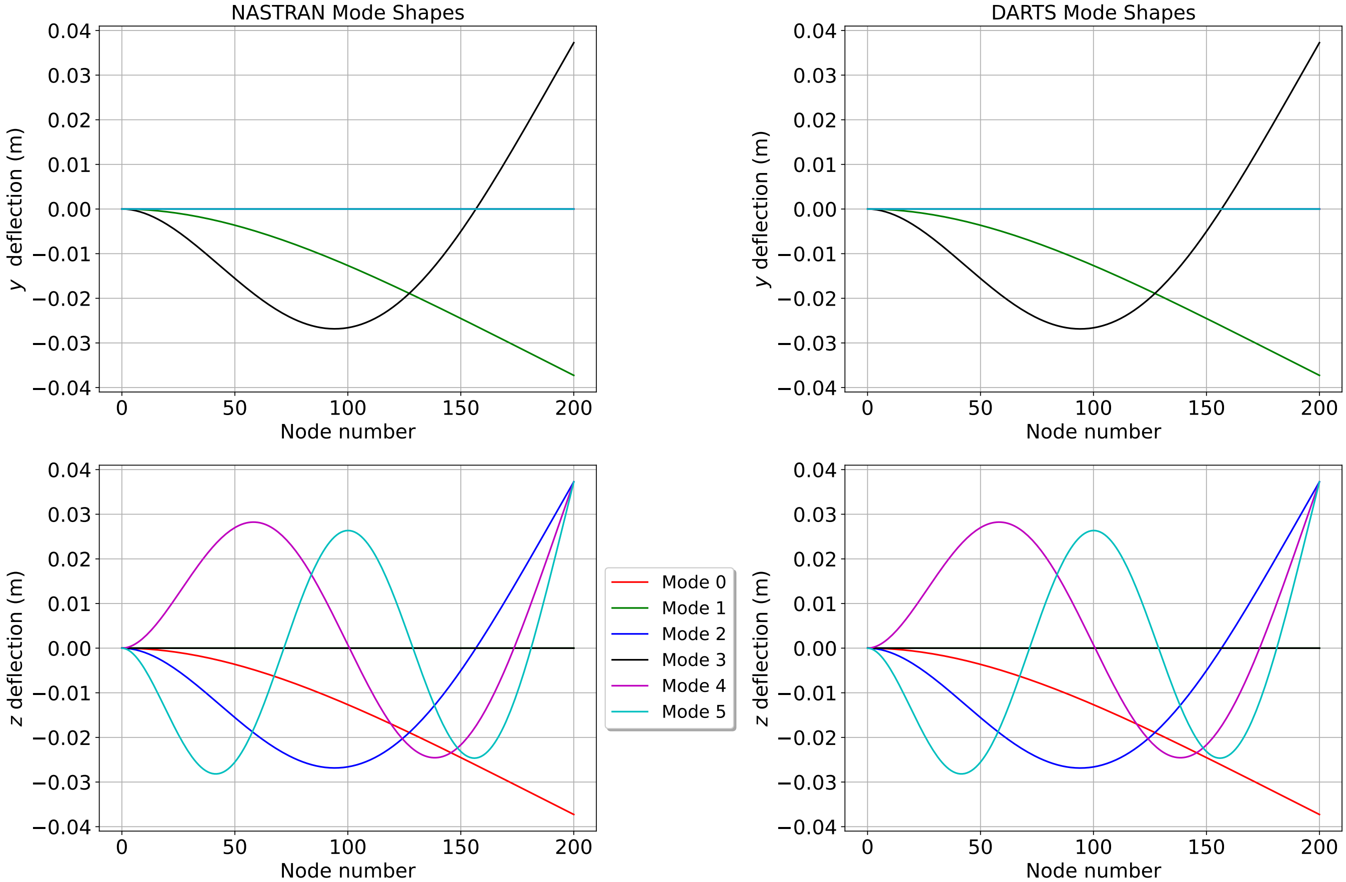


Figure 7: System mode shapes NASTRAN (left column) vs. DARTS (right column) for a double-length cantilevered beam

ensuring that the static response of system (3) is sufficiently close to system (1), and that system (3) is closer to (1) than (2). While the previous test verified the correct calculation and transfer of data, this test checks the impact of the residual vectors. Figure 10 shows the results of these two tests. The DARTS static solution with residual vectors is closer to the NASTRAN solution than without residual vectors. This validates that the residual vectors were computed correctly and transferred to DARTS correctly.

6. CONCLUSION

Reference [1] showed that deriving G&C models that include flexible bodies by incorporating flexible multibody dynamics as an intermediate step provides significant benefits over the conventional approaches. However, creating the necessary flexible body models for the flexible multibody dynamics simulation is a challenge. The FModal pipeline addresses this challenge by providing an easy-to-use toolset that allows one to quickly incorporate and iterate on flexible body models for dynamics simulations without having to be an expert in structures modeling. Even typically-neglected effects such as modal integrals can be included by simplify modifying an argument passed to a Python function. Moreover, the pipeline is general, i.e., it can be applied to an arbitrary NASTRAN model. The generated HDF5 file can be used by any multibody dynamics tool. In addition to describing the tool, some of the numerical studies used to validate the tool have been discussed. The FModal pipeline is in use by a range of projects whose FEM models vary from small kilobyte-sized models of simple beams to large spacecraft models whose FEM models are hundreds of gigabytes large.

Some additional features currently under development are described here. Thus far, validation tests for flexible body closure constraints have only been done with single-point constraints. Future work will validate multi-node closure constraints between flexible bodies. Section 3 discussed separating lumped masses connected via RBAR elements. However, lumped masses can also be connected via RBE elements, i.e., RBE1, RBE2, or RBE3. Adding support for lumped masses connected via RBE elements requires multi-node flexible body closure constraints. Hence, once multi-node flexible body closure constraints are validated, support for lumped masses connected via RBE elements can be added to FModal. Geometric stiffening due to inertial loads is another planned feature. A method [11], [12] for incorporating geometric stiffening due to inertial loads is being implemented within the FModal pipeline. Testing and validation of this feature is currently in progress and will be reported in the future.

ACKNOWLEDGMENTS

The authors would like to acknowledge the pioneering and significant contributions of Aaron Schutte [1] which form the basis for several key results reported in this paper. In addition, the authors would like to thank Reuben Isaac for adding sup-

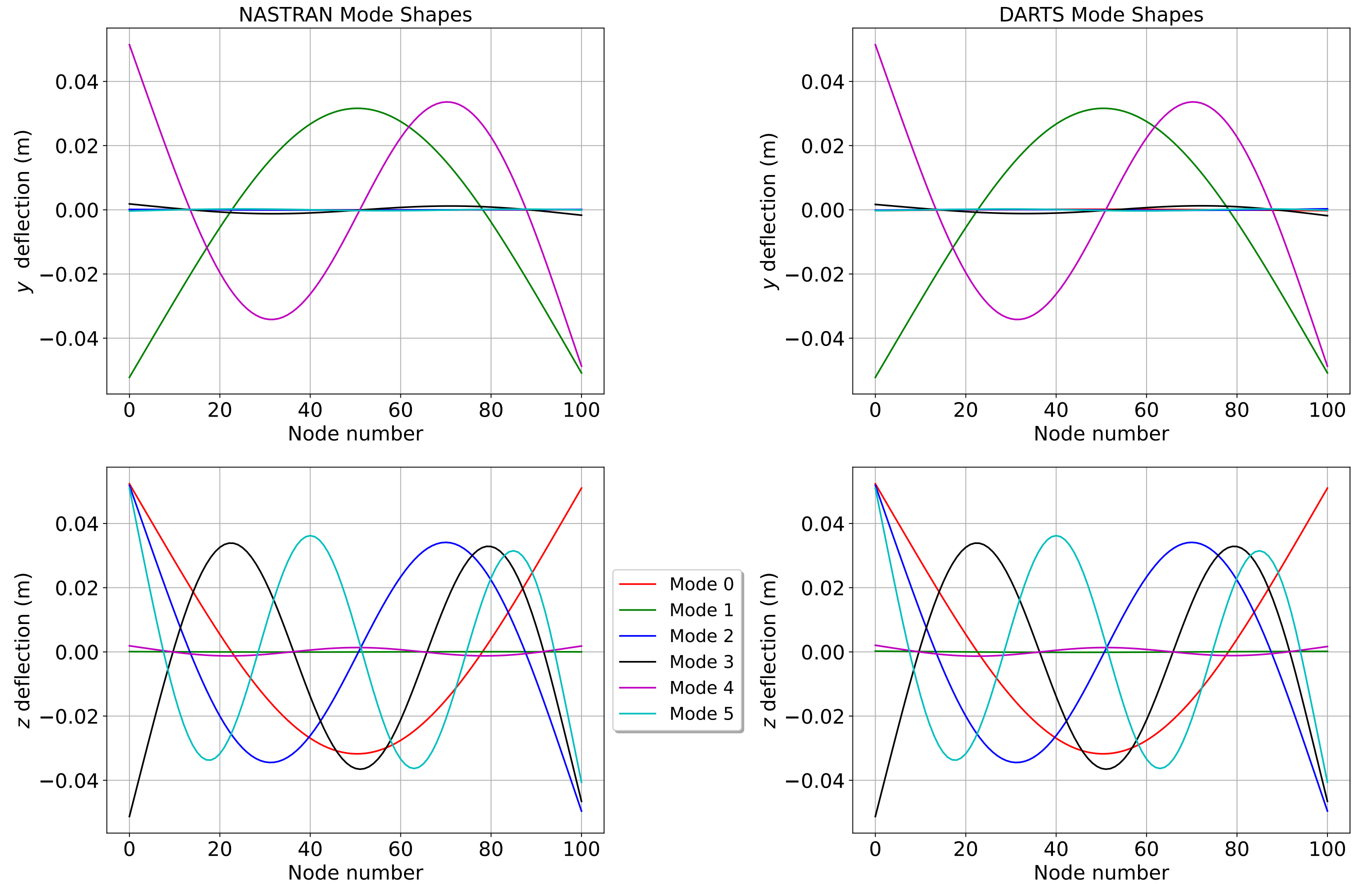


Figure 8: System mode shapes NASTRAN (left column) vs. DARTS (right column) for a free-free beam with a detachable lumped mass

port for residual vectors to FModal and for his contributions to the validation efforts. The research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

APPENDICES

A. RESIDUAL VECTOR DESCRIPTION

This appendix describes the concept of residual vectors and how they can help recover the static solution when a truncated set of modes is used. The FModal tool directly utilizes NASTRAN's residual vector procedures whose implementation may differ from the derivation shown here.

To understand how residual vectors work, first consider the nodal flexible body system [13]

$$M\ddot{x} + Kx = Pr(t) \tag{2}$$

where M is the mass matrix, K is the stiffness matrix, and x are the nodal degrees of freedom. Typically, the right side of this equation contains a single term for the forces; however, here, the forces are split up into their directions, P, and magnitudes $r(t)$. This is convenient as FEM models often apply forces on specific nodal DoFs; this information is captured in the columns of P. The dynamic information, the changing magnitude of the force over time, is captured by $r(t)$.

This equation can be written in modal form using the transformation $\bar{x} = \Pi\eta$. In this transformation, Π are a truncated set of modes, so the $\bar{x}$ deformations are a subspace of the x deformations, and thus, cannot represent all possible x deformations.

$$\begin{aligned} \Pi^T M \Pi \eta + \Pi^T K \Pi \eta &= \Pi^T P r(t) \\ I\eta + \zeta\eta &= \Pi^T P r(t) \end{aligned}$$

In the above, recall that the modes are normalized with respect to the mass matrix.

Next, pre-multiply this system by $M\Pi$ to transform the modal forces into physical forces (see [13] and Appendix A of [14]) and simplify

$$M\ddot{\bar{x}} + K\bar{x} = M\Pi\Pi^T P r(t).$$

Comparing this with the original system (Eq. 2), notice that the differences are the use of the barred quantities ($\bar{x}$ vs. x) and the forces on the right-hand side. Solving for the static solution of each of these systems yields the following:

$$\begin{aligned} x &= K^{-1} P r(t) \\ \bar{x} &= K^{-1} M \Pi \Pi^T P r(t) \end{aligned}$$

The difference between the two static solutions is

$$\begin{aligned} x - \bar{x} &= K^{-1}\left(P - M\Pi\Pi^T P\right) r(t) \\ &= R r(t) \end{aligned}$$

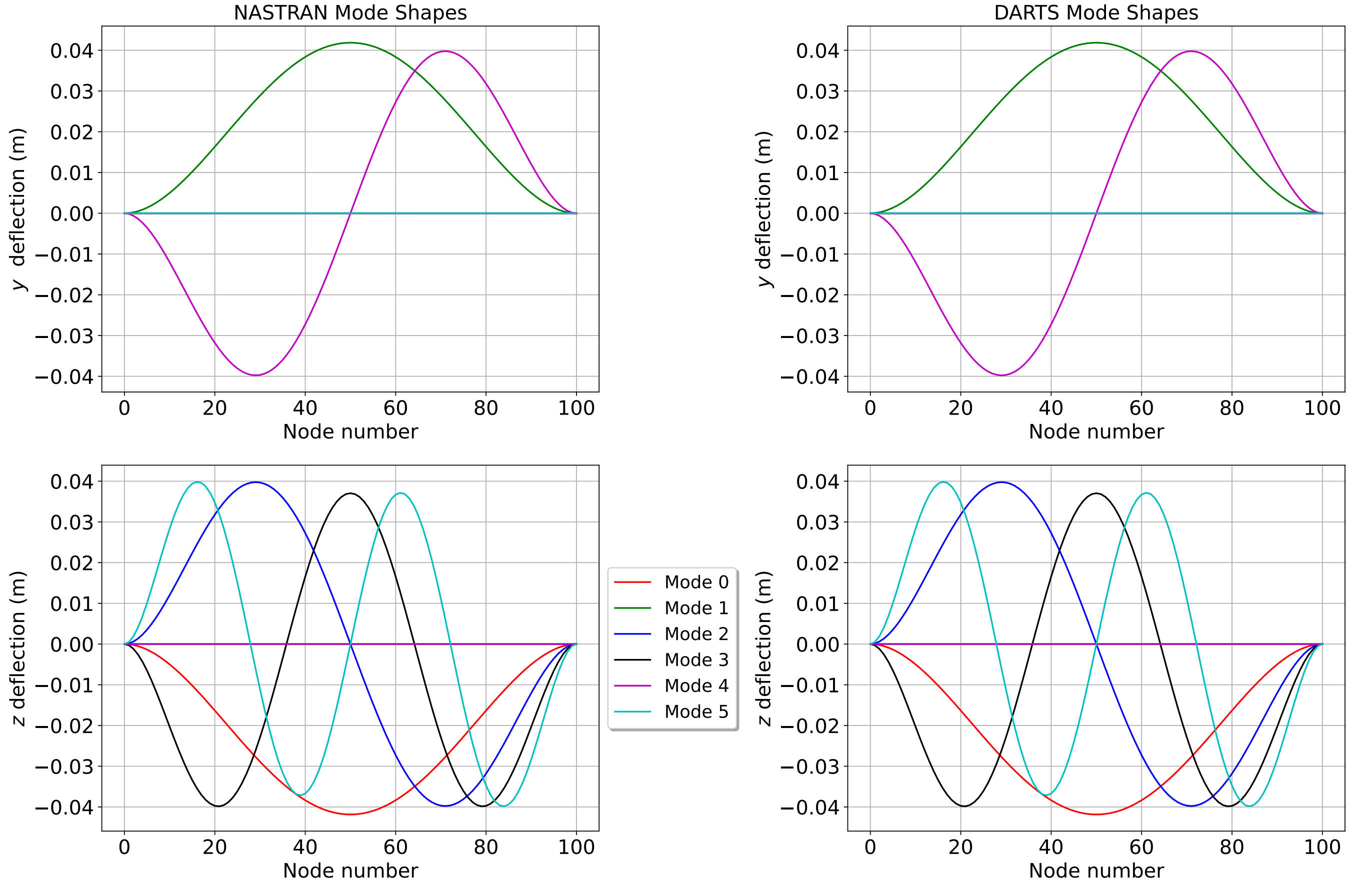


Figure 9: System mode shapes NASTRAN (left column) vs. DARTS (right column) for a fixed-fixed beam

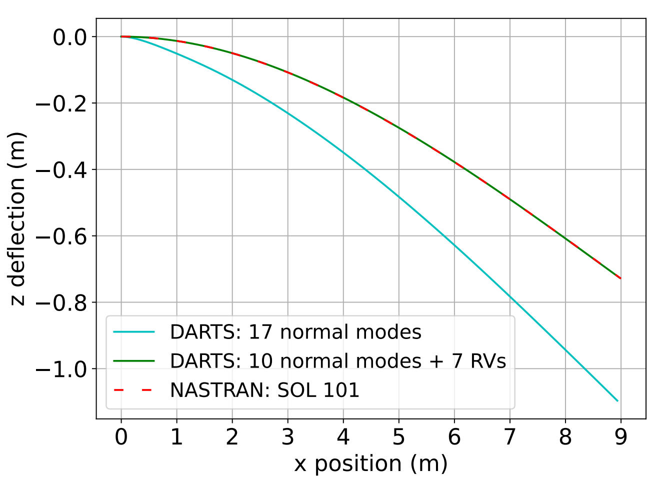


Figure 10: Static response of DARTS vs. NASTRAN with and without residual vectors

where $R = K^{-1}\left(I - M\Pi\Pi^T\right)P$. R is the residual static response, i.e., it maps the external forces to the difference between the static solution of the full and truncated systems.

In other words, appending R to the original list of mode shapes would allow one to recover the linear static solution of the original, full DoF system, while only adding a number of DoFs equal to the number of columns in P. However, in its current form, R cannot be appended to the list of mode shapes. The reason is that it lacks the required properties of the mode shapes, namely (1) they diagonalize the stiffness matrix and the diagonal entries are equal to the square of the modal frequencies and (2) they are normalized with respect to the mass matrix such that $\Pi^T M \Pi = I$.

Notice that to be able to recover the linear static response of the original full DoF system, one does not necessarily need to append R. Rather, one could append $R\Gamma_R$, where Γ_R is any full-rank matrix. This is because $R\Gamma_R$, where Γ_R is full-rank, will still span the same column space as the original R. This problem looks similar to the normalized component modes used earlier, where R takes the place of the Craig-Bampton transformation and Γ_R takes the place of the Craig-Bampton mode shapes. Moreover, solving the problem in this way will yield mode shapes that have the desired diagonalization and normalization properties. Therefore, one solves the typical eigenvalue/eigenvector problem for the following system

$$R^T M R \Gamma_R \eta_R + R^T K R \Gamma_R \eta_R = R^T P r(t)$$

i.e.,

$$\omega_R^2 R^T M R \Gamma_R = R^T K R \Gamma_R$$

where ω_R^2 are the eigenvalues and Γ_R are the eigenvectors. Then, the residual vectors of the system are $\Pi_R = R\Gamma_R$ and the associated frequencies are ω_R. These can be appended to the original set of normalized component mode shapes, and this new set of mode shapes will satisfy the two required properties mentioned earlier. Moreover, this new set of modes will be able to recover the linear, static solution of the original full DoF system.

B. Example Nastran Run Script

This appendix contains an example Python script that uses the `Nastran` class to compute the first 30 flexible mode shapes/frequencies and modal integrals of a cantilevered beam. The results are stored in an HDF5 file that can be used to create a DARTS flexible body via the `ReadHDF5Flex` class.

```python
""" This script creates the SimpleBeamFixed.h5 file. """

import os
from FModal.Nastran import Nastran

nModes = 30

fileNameH5 = "SimpleBeamFixed.h5"
fileNameDat = "SimpleBeamFixed.dat"

nast = Nastran()
nast.read(fileNameDat) # Read in the DAT file
nast.executable_path = os.getenv("MSC_NASTRAN_2019_PATH") # Set the NASTRAN executable path
nast.run_103(n_modes=nModes, aset={1:123456}) # Add node one to the a-set since it will be fixed in the multibody dynamics simulation
nast.load_op2(modal_integral_flag=True) # Enable modal integrals
nast.write_hdf5(fileNameH5) # Create the HDF5 file
```

C. Single-Length Beam DAT File Description

This appendix describes the single-length beam NASTRAN file used in many of the validation scripts. The validation scripts that use a double-length beam use two of these single-length beams connected together.

The single-length beam NASTRAN file consists of 101 equally spaced grid points along the x-axis from 0 to 9 meters. An example of one of these grid points is shown below:

```
GRID, 1, 0, 0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 0
```

Each adjacent grid point is connected by a `CBEAM` element. The `CBEAM` elements are all identical except for the grid points they connect to. An example of one of these elements connecting grid points 1 and 2 is shown below:

```
CBEAM, 11, 50002,    1,    2, 0.0000E+00, 1.0000E+00, 0.0000E+00
, 0,   0, 0.0000E+00, 0.0000E+00, 0.0000E+00, 0.0000E+00, 0.0000E+00, 0.0000E+00
```

Each `CBEAM` element uses a `PBEAM` property card (number 50002) in its description, and this `PBEAM` property card utilizes a `MAT1` material card (number 1) in its description. The `PEAM` and `MAT1` cards used are shown below.

```
$*
$*  MATERIAL CARD
$*
MAT1,    1, 2.000+9, 775193798.449612, , 8000.0, 12.000-6, 21.85000, 0.0

$*
$*  PROPERTY CARD
$*
PBEAM, 50002, 1, 2.000E-02, 6.6666667E-05, 1.6666667E-05, 0.0000E+00, 8.33333333E-05, 0.0000E+00
, 0.0000E+00, 0.0000E+00, 0.0000E+00, 0.0000E+00,  0.0000E+00, 0.0000E+00,  0.0000E+00,  0.0000E+00
, 0.0000E+00, 0.0000E+00, 0.0000E+00, 0.0000E+00,  0.0000E+00, 0.0000E+00,  0.0000E+00,  0.0000E+00
, 0.0000E+00, 0.0000E+00, 0.0000E+00, 0.0000E+00,  0.0000E+00, 0.0000E+00,  0.0000E+00,  0.0000E+00
```

D. Sample FModal HDF5 Output File

This section lists content from an example HDF5 output file produced by FModal. The groups of the HDF5 file associated with the DAT file described in Appendix C with the first node placed in the a-set are shown in Figure 11. Some of the groups (e.g., GridPosition) have not been expanded due to space limitations.

References

[1] A. S. Schutte, A. Jain, J. R. Brouwer, T. S. VanZwieten, and K. A. Simpson, "Development and verification of a pipelinefor modeling flexible multibody dynamics," NESC Technical Memo, Tech. Rep. NASA/TM-2020-5008164, NESC-RP-18-01312, August 2020. [Online]. Available: https://ntrs.nasa.gov/citations/20205008164

[2] A. Jain, *Robot and Multibody Dynamics: Analysis and Algorithms*. New York: Springer, 2011. [Online]. Available: http://www.springerlink.com/content/978-1-4419-7266-8/contents/

[3] R. Featherstone, *Rigid Body Dynamics Algorithms*. Springer Verlag, 2008.

[4] A. Schutte and F. Udwadia, "New approach to the modeling of complex multibody dynamical systems," *Journal of Applied Mechanics*, vol. 78, 2011.

[5] F. Udwadia and A. Schutte, "Equations of motion for general constrained systems in lagrangian mechanics," *Acta Mechanica*, vol. 213, pp. 111–129, 2010.

Figure 11: Sample HDF5 file output as viewed in HDFView

[6] A. Shabana, *Dynamics of Multibody Systems*. Cambridge University Press, 2013.

[7] M. C. C. Bampton and J. R. R. Craig, "Coupling of Substructures for Dynamic Analyses," *AIAA Journal*, vol. 6, no. 7, pp. 1313–1319, 1968.

[8] "The hdf group," https://www.hdfgroup.org/, accessed: 2020-01-21.

[9] A. Jain, "Darts - multibody modeling, simulation and analysis software," *Computational Methods in Applied Sciences*, vol. 53, 2020.

[10] "Dynamics and Real-Time Simulation (DARTS) Lab," \url{http://dartslab.jpl.nasa.gov/}, 2019. [Online]. Available: http://dartslab.jpl.nasa.gov

[11] A. K. Banerjee and J. M. Dickens, "Dynamics of an arbitrary flexible body in large rotation and translation," *Journal of Guidance, Control, and Dynamics*, vol. 13, no. 2, p. 221–227, Mar 1990. [Online]. Available: https://arc.aiaa.org/doi/10.2514/3.20540

[12] A. Banerjee, *Flexible Multibody Dynamics: Efficient Formulations and Applications*. Wiley, 2016.

[13] P. A. Blelloch, J. Dickens, A. Majed, and J. Sills, "Application of modal truncation vectors to the mixed and fixed boundary dynamic math models," in *AIAA Scitech 2019 Forum*. San Diego, California: American Institute of Aeronautics and Astronautics, Jan 2019. [Online]. Available: https://arc.aiaa.org/doi/10.2514/6.2019-0495

[14] J. Dickens and A. Stroeve, "Modal truncation vectors for reduced dynamic substructure models," in *41st Structures, Structural Dynamics, and Materials Conference and Exhibit*. Atlanta,GA,U.S.A.: American Institute of Aeronautics and Astronautics, Apr 2000. [Online]. Available: https://arc.aiaa.org/doi/10.2514/6.2000-1578

BIOGRAPHY

Carl Leake *is a Robotics Technologist who works in the DARTS Lab at NASA's Jet Propulsion Laboratory. He earned a B.S. in aerospace engineering from Embry-Riddle Aeronautical University in 2017 and a Ph.D. in aerospace engineering from Texas A&M University in 2021.*

Abhinandan Jain *is a Senior Research Scientist at the Jet Propulsion Laboratory, California Institute of Technology. He has been conducting research on advanced theoretical and computational methods for multibody dynamics and has authored a book and several publications in this area. He also leads JPL's DARTS Lab in the development of high-fidelity modeling and simulation tools for planetary spacecraft and autonomous robots. These tools have been in use by several NASA missions including Cassini, Mars landers as well as the Ingenuity Mars helicopter.*