

# Flexible Bodies with COTS Control Design and Physical Modeling Tools

Jason Ghidella<sup>1</sup>, Dallas Kennedy<sup>2</sup>, and Steve Miller<sup>3</sup>  
*The MathWorks, Inc. Natick, MA, 01760*

**A common problem in aerodynamic and astrodynamic engineering is accurate modeling of systems that combine flexible and rigid bodies with a controller. Mechanical and control simulators typically represent only rigid bodies and require preliminary transformation of flexible body problems into a suitable approximation. Analytical methods and commercial off-the-shelf (COTS) flexible-body analysis tools can fill this gap. Easily available COTS multibody simulation packages can efficiently simulate rigid-body machines. They can also simulate mixed rigid-flexible systems if and when they can make dynamic use of flexible-body response data from the variety of available COTS finite-element analysis programs.**

## I. Introduction

THE ability to construct high quality mechanical models often plays an important role in the design of control systems. Frequently the mechanics being modeled include flexible parts. Because of its computational expense and conceptual difficulty, many mechanical simulators do not offer native support for flexible-body modeling. In such tools, none of the simulated machine parts is assumed to change its shape or mass distribution. Modeling mixed flexible and rigid body motion is an important requirement in aerospace, automotive, and civil engineering, among other applications. It is especially important in flight simulation where aircraft structures are bent by aerodynamic loads.

In this paper, we explore two distinct mathematical approaches to modeling flexible bodies, each of which has direct application to COTS modeling tools: the *lumped-parameter method* and the *finite-element analysis (FEA)/state-space method*. These methods break the flexible-body modeling down into pieces that are less computationally demanding and that can be easily incorporated into rigid-body simulations. Both methods assume that beam deflection is small and in the linear regime. Both methods can simulate flexibility with fidelity sufficient for many applications, such as plant analysis and control design. The two methods differ in their computational expense; how open they are to greater mathematical rigor; and, in the FEA case, the need for additional COTS FEA software.

The one-dimensional bending beam is the starting point for modeling flexible bodies in physics and engineering. The lumped-parameter method, best suited for modeling beam-like geometries, discretizes the flexible body into a series of constituent or beam elements, each pair of which is linked by a spring-damper that models the flexibility between them. In its simplest version, the lumped-parameter method makes the spring moment between neighboring elements a function of local beam deflection only. While not a correct representation of the bending moment, this type of lumped-parameter approximation is useful and easy to implement, allowing a quick determination of gross flexible behavior. The method can be improved for greater accuracy and turned into a controlled approximation that converges on the correct continuum behavior. However, such improvements require more complex modeling and greater effort.

For a more rigorous approximation that can be progressively refined and made mathematically convergent, it is better to turn to the FEA/state-space method. The finite-element analysis approach, in contrast to the lumped-parameter method, makes use of the vibration analysis output of FEA programs to model flexible bodies in motion relative to a rigid-body machine. The FEA real-space discrete mesh is transformed by such analysis into a modal, frequency-response state-space that can be embedded in a larger

---

<sup>1</sup> Senior Team Lead, Technical Marketing, 3 Apple Hill Drive, Natick, MA, 01760, AIAA member.

<sup>2</sup> Senior Technical Writer, Documentation, 3 Apple Hill Drive, Natick, MA, 01760.

<sup>3</sup> Technical Marketing Manager, 3 Apple Hill Drive, Natick, MA, 01760.

mechanical control model. While computationally more intensive and conceptually more advanced, this method has major advantages: it is a correctly defined and controlled approximation and the resulting state space can be truncated or expanded as needed to model modes of interest for control design while excluding unimportant modes.

In this paper, we use the COTS software tools Simulink® and SimMechanics from The MathWorks to demonstrate how to model deflecting beams. We also discuss the merits and implementation details of each approach. SimMechanics, a multibody simulation package, can efficiently simulate mechanical systems. Used together with other MathWorks products, SimMechanics allows engineers to simulate physical plants and control systems in a single environment, apply basic and advanced control design techniques, and convert parts of models or entire models to code. Simulink and SimMechanics are easily available COTS tools based on MATLAB®. It will be shown that these tools provide an efficient environment for simulating mixed flexible- and rigid-body mechanical systems, especially in control systems applications.

This paper will first discuss the lumped-parameter method and then the finite-element analysis approach, which is better suited to three-dimensional geometries and (in combination with state-space methods) more applicable to control design. A comparison is made between the two methods and their corresponding results from modeling a simple bending beam, highlighting both the strong points and the limitations of the each approach and how to address them.

## II. The Lumped-Parameter Method

The lumped-parameter approach approximates a flexible body as a set of coupled rigid bodies. It can be implemented by a chain of alternating bodies and joints actuated by springs and dampers. The spring stiffness and damping coefficients are functions of the material properties and the geometry of the flexible elements. While lumped-parameter methods can be extended to more complicated systems, they are best suited for systems with chain-like geometries.

### A. Generalized Beam Elements and Degrees of Freedom

In this section we apply the lumped-parameter method first to a simple beam and then to a flexing cantilever. A beam is represented by a series of *generalized beam elements* (GBEs), each of which is a body-joint-body combination. This formulation actuates and senses the joints directly as relative *degrees of freedom* (DoFs).

The lumped-parameter method involves five steps:

- 1) Divide the beam into discrete elements and determine the DoFs of each element.
- 2) Represent the DoFs with a joint acting at the middle of each element along the neutral axis (the line through the element that suffers no stretching or compression).
- 3) Use flexible body theory to determine the effective spring constants from the geometry, material properties, and boundary conditions.
- 4) Apply damping as necessary to each DoF.
- 5) Couple the GBEs with welds.

Simulink and SimMechanics simplify the task of deriving lumped-parameter approximations by enabling engineers to:

- Wrap the GBE model into a prototype masked subsystem
- Turn this subsystem into a library block that can be instantiated in a model wherever and as often as necessary
- Organize all GBE data into data structures
- Implement the body-joint-body-joint... structure by connecting a coordinate system on each side of each GBE to its immediate neighbor (adjoining coordinate system)

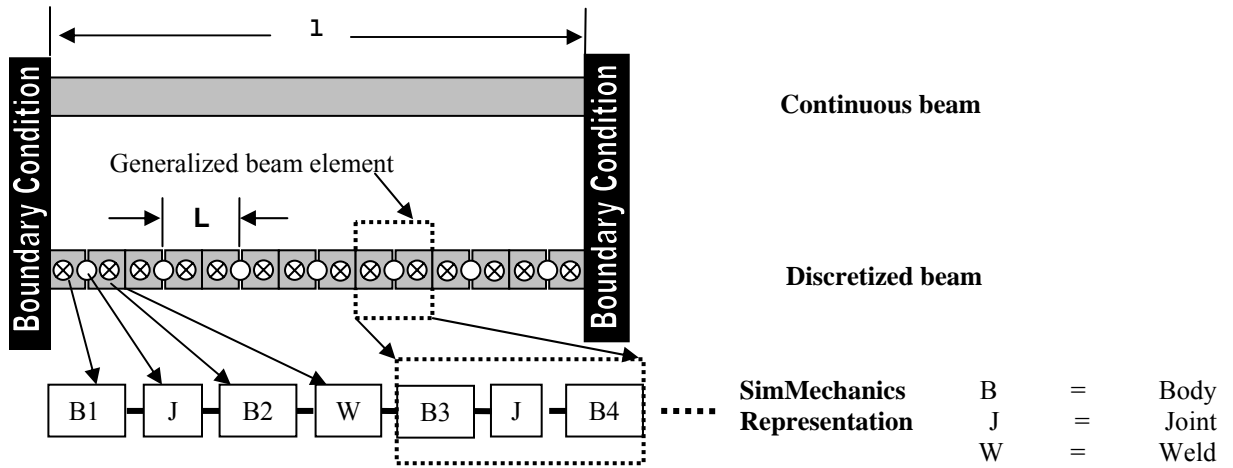


Figure 1. Discretization of a beam into generalized beam elements.

## B. Applying the Lumped-Parameter Method to Generalized Beam Elements and a Bending Beam

To analyze the GBE motion and forces, define  $\mathbf{X}$  and  $\mathbf{F}$  to be generalized coordinates and forces at one end of the GBE. Assume that the other end is temporarily fixed. Let  $\mathbf{x}$  represent the joint motion. Then

$$\mathbf{X} = \mathbf{g}(\mathbf{x}), \quad d\mathbf{X} = d\mathbf{g}(\mathbf{x}) \cdot d\mathbf{x} = [\mathbf{J}] \cdot d\mathbf{x}, \quad \text{where } [\mathbf{J}] = d\mathbf{g}/d\mathbf{x} \quad (1)$$

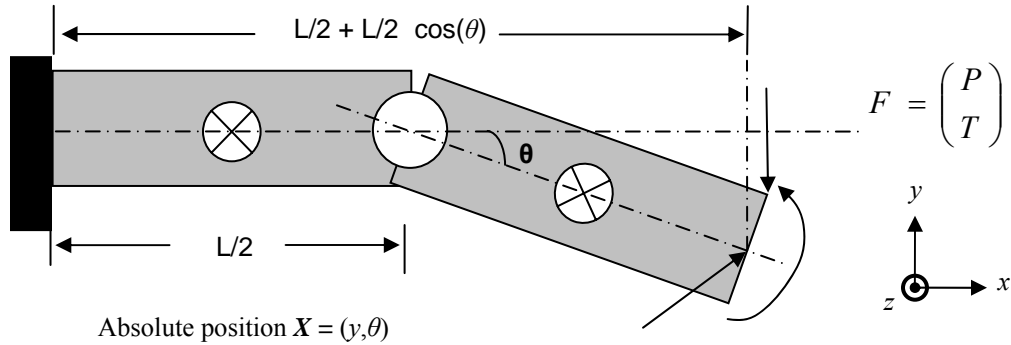
The generalized force  $\mathbf{F}$  at the tip and generalized force  $\mathbf{f}$  at the joint can be expressed with a generalized stiffness matrix  $[\mathbf{K}]$  or  $[\mathbf{k}]$ :

$$\mathbf{F} = [\mathbf{K}] \cdot d\mathbf{X}, \quad \mathbf{f} = [\mathbf{k}] \cdot d\mathbf{x} \quad (2)$$

The Jacobian  $[\mathbf{J}]$  relates the two matrices, such that  $\mathbf{f} = [\mathbf{J}]^T \cdot [\mathbf{K}] \cdot [\mathbf{J}] \cdot d\mathbf{x}$ . Thus:

$$[\mathbf{k}] = [\mathbf{J}]^T \cdot [\mathbf{K}] \cdot [\mathbf{J}] \quad (2a)$$

Now we apply this technique to a cantilever or beam bending in a plane. In this example, one revolute represents the bending of a single GBE relative to its neighbor.



**Figure 2. Generalized beam element forces and deflection.**

$P$  and  $T$  are the force and moment (torque), respectively, at the end of the GBE. The moment  $\tau$  at the joint can be expressed from equation (2a) as

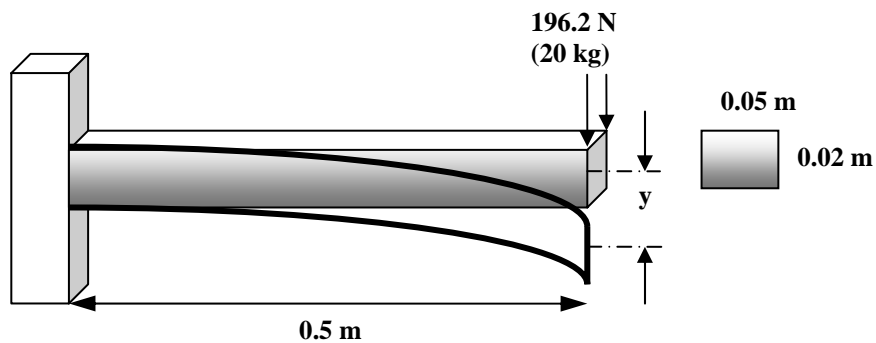
$$\tau = k \cdot d\theta, \text{ where } k = (E \cdot I_{Az})/L \quad (3)$$

in terms of the area moment of inertia  $I_{Az}$  (second moment of area of the beam's  $yz$  cross-section) and Young's elastic modulus  $E$ . The rotational joint  $n$  executes damped oscillations according to the normalized moment equation

$$d^2\theta/dt^2 + 2\zeta_0\omega_0 \cdot d\theta/dt + \omega_0^2\theta = \tau_{\text{external}}, \text{ where } \omega_0^2 = k/I_{Mz} \quad (4)$$

$\omega_0^2$  is the ratio of the effective spring constant  $k$  and the mass moment of inertia  $I_{Mz}$ , both assumed to be the same for all GBEs. The damping coefficient  $2\zeta_0\omega_0$  that multiplies the velocity is a quasiempirical value that accounts for energy lost to viscoelastic effects.

The beam is made of an aluminum alloy of density  $2700 \text{ kg/m}^3$  with a Young's modulus of  $6.9 \times 10^{10} \text{ N/m}^2$ . We apply a load of  $196.2 \text{ N}$  (equivalent to the weight of  $20 \text{ kg}$ ) to the tip.

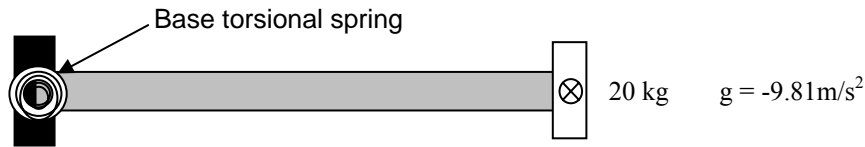


**Figure 3. Beam or cantilever bending in the  $xy$  plane.**

Calculation yields these values, neglecting the beam weight:

Mass (analytic)	1.35 kg
Equilibrium deflection at tip (analytic)	-3.56 mm
Equilibrium deflection at tip (lumped GBEs)	-3.54 mm
Vibration frequency (lumped GBEs)	63.7 Hz

Now we apply the lumped-parameter method with ten GBEs to the same cantilever, mounted to a wall with a torsional spring and including the beam weight. Again, we apply the 20 kg-mass-equivalent load to the tip. The beam bends by this load as well as by its own weight (bending under its own weight is a *rigid body mode* of the beam). The addition of the beam weight reduces the vibration frequency by a factor of about 10. This simulation uses a model available from MATLAB Central.<sup>1</sup>



Base spring stiffness $K_s$	10,000 N/m
Equilibrium deflection at tip (lumped GBEs)	-3.55 mm
Vibration frequency (lumped GBEs)	6.82 Hz

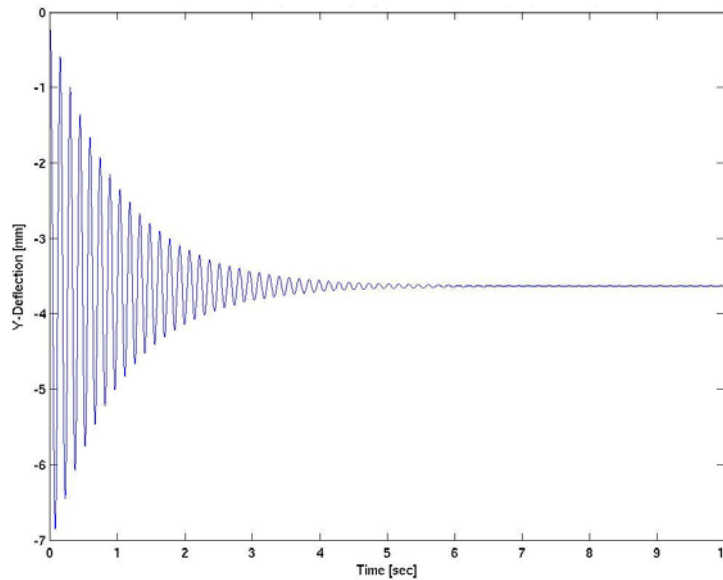


Figure 4. Motion of a spring-mounted, body-loaded beam, modeled by SimMechanics with 10 GBEs.

### C. Limitations and Corrections to the Lumped-Parameter Method

The lumped-parameter method outlined here uses purely local forces as functions of local deflections. This approximation yields important qualitative and semi-quantitative insights and is useful for estimating the gross behavior and endpoint deflection of the beam. It is not, however, useful for modeling local dynamics along the beam or the beam's vibrational modes. The restoring moment generated by bending the cantilever is proportional to its curvature. This method approximates the curvature by the local difference in slopes rather than by the correct nonlocal difference in slopes between neighboring GBEs, which cannot

be represented in a series of independent elements. Thus, while the lumped-parameter method is a convenient way to represent a flexible body, it does not, in general, yield correct results and cannot be made progressively more accurate by refining the discretization.

This flaw can be fixed by correctly approximating the local curvature and implementing this correct form in the local bending moment. Let  $\alpha_n$  be the absolute deflection angle of GBE(n), and  $\theta_n$  be the relative deflection angle of GBE(n) with respect to GBE(n - 1):  $\theta_n = \alpha_n - \alpha_{n-1}$ . Then to linear order,

$$(\text{Bending Moment})_n \sim (\text{Curvature})_n \approx (d^2y/dx^2)_n \sim (dy/dx)_{n+1} - (dy/dx)_{n-1} \sim \alpha_{n+1} - \alpha_{n-1} = \theta_{n+1} + \theta_n$$

The bending moment at GBE(n) depends on the relative deflection at GBE(n+1) and GBE(n), not just GBE(n). This result is equivalent to the standard partial differential equation for beam deflection, second-order in time, fourth-order in space, in terms of vertical displacement  $y$ . One converts moment to force by taking another derivative and obtains the *net* force on the GBE by taking one more derivative. The result is fourth-order in  $y$ . These steps result in a consistent discrete approximation that can be successively refined to converge to the correct beam moment and deflection. The drawback is that each GBE moment is no longer independent of its neighbors.

### III. The Finite-Element Analysis Approach

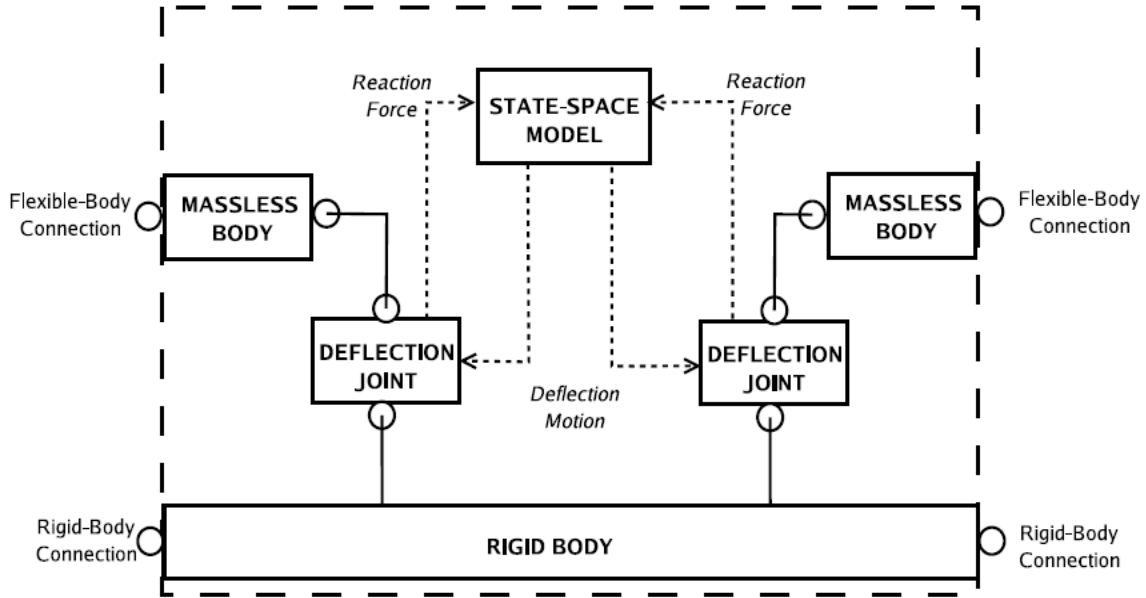
To obtain more accurate and useful results, we can incorporate the *frequency* or *modal response* of the flexible bodies under deformation, as calculated by a finite-element analysis application, into a SimMechanics model; use this response to calculate the deflections of the flexible body; and then superimpose those deflections on the rigid-body motion. This approach is not only more accurate, it is also more easily applied to complicated geometries than is the GBE, lumped-parameter method. Perhaps the most important advantage of this method is the ease of trading off between the level of detail (fidelity) and the computational intensity (cost) of the simulation. This is crucial for control design applications, where a lower level of fidelity is acceptable and faster model execution is critical.

#### A. Finite-Element State-Space, Coordinate Systems, and Degrees of Freedom

The FEA approach involves six steps:

1. Identify the FEA degrees of freedom and corresponding SimMechanics coordinate systems for the load forces.
2. Identify the FEA degrees of freedom and corresponding SimMechanics coordinate systems for the deflections.
3. Perform the finite-element analysis and extract the modes.
4. Construct the state-space representation of the flexible body dynamics.
5. Feed the correct forces into the state-space model. These are either externally imposed forces or reaction forces from neighboring bodies.
6. Use the output of the state-space representation to motion-actuate the appropriate SimMechanics joint primitives connected to the deflecting coordinate systems. The joint primitives are motion-actuated by a “black box” subsystem whose output is the deflection calculated from the FEA results.

These steps superimpose the deflections of the flexible body on top of the motion of the rigid body. Figure 5 shows how the deflections are superimposed.



**Figure 5. Embedding a state-space flexible body model into a SimMechanics model.**

Simulink and SimMechanics simplify the task of using frequency or modal response by enabling engineers to:

- Use Body coordinate systems, Joint primitives, and actuators to cleanly separate the flexible-from the rigid-body motions.
- Represent the flexible-body motion in a convenient and useful state-space form.
- Select the desired level of detail by including only the states necessary for the analysis.
- Fine-tune the trade-off between fidelity and speed or cost.

The most direct way to implement the black box in Simulink is to use a State-Space block or an LTI System block. Either block implements the flexible-body dynamics in a state-space model and uses the results to actuate the motion of massless bodies to which other SimMechanics blocks are connected. You can use this method to model multiple flexible bodies and complex systems comprising many flexible parts, if the deformations remain small enough for linear approximation.

## B. State-Space and the Damped, Coupled, Linear Oscillator Form

The state-space model is based on an abstract, multi-DoF version of Eq. (4),  $M\ddot{q} + C\dot{q} + Kq = Fu$ , with  $M$ ,  $C$ ,  $K$ , and  $F$  as mass, friction, spring, and force matrices acting on the joint DoFs  $q$  and system inputs  $u$  (for example, reaction forces). A proper choice of coordinate transformation  $\Phi$  from the spatial DoFs  $q$  to the *normal coordinates*  $\eta = \Phi \cdot q$  yields the normalized form of the dynamical equations,  $\ddot{\eta} + 2\Gamma\dot{\eta} + \Omega\eta = \Sigma u$ , and diagonalizes  $\Omega$ , the *stiffness matrix* whose eigenvalues are the squared angular frequencies,  $\omega_n^2$ . The  $\eta$ ,  $\Phi$ , and  $\Omega$  values are obtained from the FEA program. Since there is no simple way to derive  $\Gamma$  from theory, we use approximations and empirical values instead. In the commonly used *proportional damping* ansatz,  $\Gamma$  is assumed to be diagonal, with each eigenvalue  $\gamma_n = \xi_n \omega_n$ . A complete state-space also includes system outputs  $y$  (for example, sensors), represented as linear functions of the DoFs and their velocities and accelerations:  $y = (Tq, T\dot{q}, T\ddot{q})$ . (Here,  $y$  here should not be confused with vertical deflection.)

We represent this collection of forced, damped, and coupled linear harmonic oscillators in linear time-invariant (LTI) state-space form

$$\begin{aligned}\dot{x} &= Ax + Bu \\ y &= Cx + Du\end{aligned}\tag{5}$$

with

$$x = \begin{pmatrix} \eta \\ \dot{\eta} \end{pmatrix}, A = \begin{bmatrix} 0 & I \\ -\Omega & -\Gamma \end{bmatrix}, B = \begin{bmatrix} 0 \\ \Sigma \end{bmatrix}, C = \begin{bmatrix} \Theta & 0 \\ 0 & \Theta \\ -\Theta\Omega & -\Theta\Gamma \end{bmatrix}, D = \begin{bmatrix} 0 \\ 0 \\ \Theta\Sigma \end{bmatrix}\tag{6}$$

where  $\Sigma = \Phi^T \cdot F$  and  $\Theta = T \cdot \Phi$ .

### C. Inducing and Curing Algebraic Loops

Embedding FEA frequency response into a SimMechanics model introduces algebraic loops. These are additional, non-time-increment steps inserted into the simulation by Simulink to solve time-independent (algebraic) constraints. Here, they arise when Simulink seeks a consistent solution for the acceleration of the massless bodies. The acceleration of the massless bodies depends upon the reaction force, which itself is a function of the acceleration — hence the algebraic loop. Algebraic loops should be avoided because they can slow down the simulation and make it less accurate.

We can break algebraic loops with Transfer Function blocks with little or no effect on fidelity. There are two ways to implement this workaround:

1. Filter each component of the output  $y$  as we have formulated it above through a transfer function of the form:

$$\mathbf{H}(\mathbf{s}) = \frac{\mathbf{K}}{\mathbf{s} + \mathbf{K}}$$

2. Filter only the position components of  $y$  through three separate transfer functions to obtain consistent positions, velocities, and accelerations that can be used in the joint actuator:

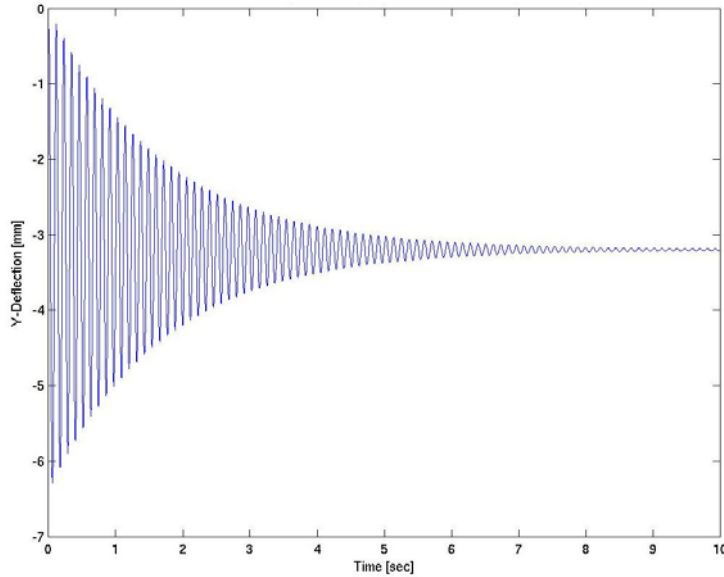
$$\mathbf{P} = \frac{\mathbf{K}^3}{(\mathbf{s} + \mathbf{K})^3} \mathbf{Y}$$

With the latter choice, we can simplify the state-space system by not producing velocities or accelerations as output. Both formulations are low-pass filters with poles at  $s = -K$  and require the somewhat arbitrary selection of a cutoff constant  $K$ . The second formulation illustrates a general approach for getting consistent position, velocity, and acceleration signals given only a position input.

Now we apply the FEA method to the same bending aluminum cantilever previously modeled using the lumped-parameter method. Without spring-mounting at the wall or any beam weight, the lowest two vibration frequencies are 69.8 Hz and 434 Hz.

Consider this cantilever with the same spring loading and revolute joint at the wall and the same beam weight as before. Modeling the cantilever using only the lowest vibration mode in FEA yields a vibration frequency and equilibrium deflection of 8.76 Hz and -3.20 mm, respectively.

*All FEA results were obtained from COSMOSWorks 2006 SP2.0, a commercial FEA software application, using default mesh parameters (global size 7.94 mm, tolerance 0.40 mm). The SimMechanics model is available from MATLAB Central; see [link](#) at the end.<sup>1</sup>*



**Figure 6. Motion of spring-mounted, body-loaded beam, modeled by FEA and state space dynamics.**

#### IV. Comparison of Methods and Results

With a reasonable number of discrete elements, the lumped-parameter method gives accurate results for static bending, but the FEA method gives more accurate results for vibration modes and frequencies.

<b>Body-loaded, spring-mounted cantilever</b>	<b>Vibration frequency (Hz)</b>	<b>Equilibrium deflection (mm)</b>
Lumped parameters (10 GBEs)	6.79	-3.63
FEA/Frequency response	8.76	-3.20

Each method has advantages and disadvantages.

- The lumped-parameter method is easy and quick to implement for flexible bodies with chain-like geometries but is more difficult and ambiguous, hence less useful, for modeling bodies with higher-dimensional geometries.
- The lumped-parameter simplification to independent GBEs incorrectly represents the curvature bending moment.
- The FEA/state-space method is straightforward to implement for any body geometry and any number of DoFs. It is also more naturally suited to control design and analysis problems.
- Once the FEA state-space is embedded into a SimMechanics model, this method introduces algebraic loops as Simulink seeks a consistent solution for the acceleration of the massless bodies. The acceleration depends on the reaction force, which is itself a function of the acceleration. Algebraic loops can slow down the simulation and make it less accurate.

Algebraic loops can be broken with Transfer Function blocks, with little or no effect on fidelity.

- For a highly refined discretization and a large number of DoFs (many GBEs and joints in the lumped-parameter case, many states in the FEA case), the dynamics become difficult to solve numerically.
- Both methods can lead to dynamics controlled by a wide range of frequencies. A large ratio of the highest important frequency to the lowest makes the system technically “stiff” and difficult to simulate, meaning that the dynamics simultaneously reflect many different time scales. In the FEA method, one can select the modes and frequencies of interest, neglecting the others, to mitigate this difficulty.

## V. Finite-Element Analysis, State-Space Methods, and Control Design

A controller is useful if it receives and responds to signals at frequencies corresponding to modes of plant motion that are to be controlled. A controller is stable if it filters out and suppresses frequencies corresponding to modes of plant motion that one wants to ignore. Although there are exceptions, low and high frequencies typically constitute the latter set, while mid-range frequencies make up the former set.

State-space methods have a natural relation to control design, because each state usually corresponds to a particular mode of the plant, with a particular frequency. We can analyze the behavior and response of our plant within a restricted range by truncating the full state space to just the modes we want, often with a significant saving in computational cost. Matrix manipulation and analysis techniques available in MATLAB then enable us to isolate the states of interest and the coupling between these states and the modes we want to ignore.

The linear time-invariant form of the state-space derived from finite element analysis is the foundation of control design for plants with flexible bodies. The  $(A, B, C, D)$  matrices of Eq. (5-6) are identical to those used to represent plant motion and to start the design and analysis of controllers. For a mixed flexible-rigid system, these matrices become embedded in a larger state space representation.

## VI. Conclusion

Flexible bodies can be modeled using COTS software tools. We have presented two of the most common methods for modeling flexible bodies based on discretization of the bodies, with the goal of achieving a balance between accuracy and computational cost. We implemented both of these methods in the Simulink environment using SimMechanics and compared the results.

The lumped-parameter method discretizes the bodies into independent elements responding to local torques. This method is conceptually simple and easy to implement, but has serious limitations. Finite-element analysis is more general and much more powerful, but requires an additional FEA software application to calculate the modes and frequencies of your flexible-body system. This method can be systematically improved to converge to physically correct results.

Simulink and SimMechanics enable easy implementation of these two methods. These applications from The MathWorks, along with COTS FEA tools, make it possible to simulate flexible bodies at the right level of fidelity for aerodynamic and astrodynamics control design, as well as many other areas of engineering.

## References

<sup>1</sup> Learn more about flexible body modeling with this text and model archive available at MATLAB Central: <http://www.mathworks.com/matlabcentral/fileexchange/loadFile.do?objectId=11027>

Klaus-Jürgen Bathe, *Finite Element Procedures*. Prentice-Hall, 1996.

S. Crandall, D. Karnopp, E. Kurtz, and D. Pridmore-Brown, *Dynamics of Mechanical and Electromechanical Systems*. McGraw-Hill, 1982.

Galileo Galilei, *Dialogues Concerning Two New Sciences* (1638), trans. H. Crew and A. di Salvio. Dover, 1956.

*Getting Started with Control System Toolbox* and *Control System Toolbox User's Guide*. The MathWorks, 2000-2007.

*Getting Started with Simulink* and *Using Simulink*. The MathWorks, 1990-2007.

Michael R. Hatch, *Vibration Simulation using MATLAB and ANSYS*. Chapman & Hall/CRC, 2001.

J. N. Reddy, *An Introduction to Finite Element Methods*. McGraw-Hill, 1993.

*SimMechanics User's Guide*. The MathWorks, 2001-2007.

©1994-2007 by The MathWorks, Inc., MATLAB, Simulink, Stateflow, Handle Graphics, Real-Time Workshop, SimBiology, SimHydraulics, SimEvents, and xPC TargetBox are registered trademarks and The MathWorks, the L-shaped membrane logo, Embedded MATLAB, and PolySpace are trademarks of The MathWorks, Inc. Other product or brand names are trademarks or registered trademarks of their respective holders.