States

'States' is a collective term for all time-dependent properties of a simulation model.

The aim of a time domain solution is to find the states' development in time, but they also play a vital role in frequency domain simulations. Together with the model geometry and parameters, which do both not change in time, the model's behavior is consistently described when the states and their change in time is known.

Usually, the term 'states' means the Modeling Element states, which are single quantities provided by Modeling Elements and handled by the Equations of Motion. These are described in State Types.

In contrast, there is also the term 'model state' or 'state of the model', which comprises the whole of all Modeling Element state values. See, in particular, Working State and Initial State.

State Types

In Simpack there are different types of Modeling Element states. All states are handled by the respective Modeling Elements mentioned in the descriptions below.

Joint (including Connections treated as Joints by the multibody formalism) s_{jnt} (position), $\dot{s}_{\mathrm{jnt}}$ (velocity).

These are one of the most common types and describe by means of Joints (and Connections) the kinematical behavior of the Bodies. They represent the generalized coordinates of the Bodies' movements. Due to requirements from the solution process, they consist of two independent values, the so-called first-order states, one for the position and one for the velocity in the respective direction. Joint states are subject to the integration over time in time domain solutions. For more information, see Joint States. The Connection states are not actually states, but their values are mapped to Joint states, which are then solved by the Solver. For elements that support referencing states, most element types allow you to assign either Connection or Joint states.

Flexible Body states s_{flx} (position), $\dot{s}_{\mathrm{flx}}$ (velocity)

describe the deformation behavior of Flexible Bodies using the modal coordinates or the nodal representation, depending on the Flexible Body type. They also consist of two independent first-order states, one for position and one for velocity, and they are also subject to the time domain integration. See also Modal Approach.

Dynamic states s_{dyn}

are used by some Force Elements and Control Elements and describe internal dynamic behavior of these elements. A dynamic state requires only one first-order state. They are also subject to the time domain integration. See States.

Algebraic states s_{alg}

are again used by some Force Elements and Control Elements but also by some Markers. The latter describe internal non-dynamic behavior of the Markers, for example contact point positions, and are called 'on position level'. The former describe dynamic behavior, e.g. force values, thus they are called 'on acceleration level'. They are usually handled by the integrator and subject to the integrator's tolerance management but they are not integrated. Some Modeling Elements also have an option to handle algebraic states internally. If they are handled by the integrator then it must be able to solve differential-algebraic equations (DAEs). See also States.

Constraint and Connection Forces and Torques s_{λ},

or Constraint(including Connections treated as Constraints by the multibody formalism) states, are a special kind of algebraic states. They contain the forces and torques invoked by Constraints (and Connections). The integrator must be able to solve differential-algebraic equations (DAEs) to handle Constraint states.

Root states s_{root}

are used by some Force Elements and Control Elements. They contain the switching status if, for example, a Force Element switches between different force laws. The switching process and the root states are handled by the Time Integration solver. However, not all integrators support this. See Root Functions, Root Functions and Root Functions.

Descriptive states s_{desc}

store intermediate values in Force Elements and Control Elements. They are used to make the intermediate results properly available in the Measurements solver and for the continuation runs (see State Initialization). The descriptive state values are not interpolated between the solver steps (see Output Point Sampling for the model states interpolation); the value of the preceding step is held constant until the end of the current step. This type of state is the only type which is not handled by the solver but by the Modeling Elements themselves. See States. See also the User Routines Descriptive States.

More information can be found in the sections describing the Solvers, and also the State Sets.

Within the solvers, all state values are handled in coherent SI units. The values entered in the model (user interface, SubVars) are converted before being passed to the solvers. See Unit Handling for more information.

Working State

'Working state' means the model state that arises from the 'current' values of the single Modeling Element states. The user sees the model in its working state in the 3D Page. The working state can be modified by, for example,

- Editing the 'State' values of Modeling Elements that possess states, see State Types
- Applying State Sets to the model
- Importing State (.spckst) files, see Exporting and Importing States, into the model
- Resetting states to zero, see Resetting States to Zero

The working state is saved with the model and reappears when a model is loaded again.

The user may also store the working state in State Sets within the model or export it to State (.spckst) files, see Exporting and Importing States.

See also Initial State.

Initial State

The initial state is the model state the Solvers begin their actual work with. Online solvers (see On- and Offline Solving) always start from the Working State. Offline solvers normally start from the model state given by the Simpack model file, which is the working state at the time the model was saved. The user may, alternatively, have the solvers start from a given State (.spckst) file, see simpack-slv (Simpack Solver).

Additionally, both online and offline solvers may automatically initialize or update some states at each solver start. See Solver Initialization for more information about the initialization and how to avoid it when concatenating solver runs (General: Concatenating Solver Runs). The initial state is the model state after this initialization.

State Vector

The first five state types listed in State Types above are collected as subvectors in the so-called 'state vector'

$$\mathbf{x}(t) = \begin{pmatrix} \mathbf{s}_{\mathrm{jnt}} \\ \mathbf{s}_{\mathrm{flx}} \\ \mathbf{s}_{\mathrm{dyn}} \\ \dot{\mathbf{s}}_{\mathrm{jnt}} \\ \dot{\mathbf{s}}_{\mathrm{flx}} \\ \mathbf{s}_{\lambda} \\ \mathbf{s}_{\mathrm{alg}} \end{pmatrix}$$

This vector is the main input to the equations of motion and is passed to the Time Integration solver, see General Nonlinear. Linear (frequency domain) solvers use a slightly modified state vector, see Linearized.

Degrees of Freedom (DOF)

Another frequently used term is 'degree of freedom', or 'DOF'. There is a well-defined relationship between the degrees of freedom and the states: On the one hand, each single Joint, Flexible Body and dynamic state provides one 'degree of freedom' (DOF) to the model. Thus one DOF corresponds to one or two first-order states (see State Types above). On the other hand, each constraint state removes one degree of freedom by constraining a movement in a particular direction. Thus the resulting number of degrees of freedom of a simulation model n_{model} can be determined by the equation

$$n_{\mathrm{DOF,model}} = n_{s_{\mathrm{jnt}}} + n_{s_{\mathrm{flx}}} + n_{s_{\mathrm{dyn}}} - n_{s_{\lambda}}$$

with $n_{s_{\mathrm{jnt}}}$ the number of Joint states s_{jnt}, $n_{s_{\mathrm{flx}}}$ the number of Flexible Body states s_{flx}, $n_{s_{\mathrm{dyn}}}$ the number of Force Element and Control Element dynamic states s_{dyn} and $n_{s_{\lambda}}$ the number of Constraint Forces and Torques s_{λ}.

The number of equations to be solved by the integrator in a time domain simulation is

$$n_{\mathrm{Eq,model}} = 2\left(n_{s_{\mathrm{jnt}}} + n_{s_{\mathrm{flx}}}\right) + n_{s_{\mathrm{dyn}}} + n_{s_{\mathrm{alg}}} + n_{s_{\lambda}}$$

where $n_{s_{\mathrm{alg}}}$ is the number of algebraic states s_{alg}. All state equations are always solved, although some degrees of freedom may be removed by Constraintsor Connections.

General Nonlinear

The Simpack solvers convert the Modeling Elements and model structure into a set of nonlinear ordinary differential equations (ODE). Models containing Constraints or Connections in closed kinematic loops, and/or algebraic states require an additional set of algebraic equations. The complete set of equations is then called differential-algebraic equations (DAE). See State Types for an explanation of the different state types.

System of Equations

The nonlinear DAE set in its most general, explicit form is as follows:

$$\begin{aligned} \dot{\mathbf{p}} &= \mathbf{T}(\mathbf{p})\,\mathbf{v} && \text{(kinematic)} \\ \mathbf{M}(\mathbf{p})\,\dot{\mathbf{v}} &= \mathbf{f}(\mathbf{p},\mathbf{v},\mathbf{c},\mathbf{s},t,\mathbf{u},\lambda) - \mathbf{G}^T(\mathbf{p},\mathbf{c},\mathbf{s},t,\mathbf{u})\,\lambda && \text{(momentum)} \\ \dot{\mathbf{c}} &= \mathbf{f}_c(\mathbf{p},\mathbf{v},\mathbf{c},\mathbf{s},t,\mathbf{u},\lambda) && \text{(dynstates)} \\ 0 &= \mathbf{g}(\mathbf{p},\mathbf{c},\mathbf{s},t,\mathbf{u}) && \text{(cnsts)} \\ 0 &= \mathbf{b}(\mathbf{p},\mathbf{v},\mathbf{c},\mathbf{s},t,\mathbf{u},\lambda) && \text{(algebraic)} \end{aligned}$$

where

- $\mathbf{p}$ are the position states of Joints (including Connections treated as Joints by the multibody formalsim), and Flexible Bodies (s_{jnt} and s_{flx})
- $\mathbf{T}$ is the transformation matrix for angles
- $\mathbf{v}$ are the Joint, Connection, and Flexible Body states on velocity level ($\dot{s}_{\text{jnt}}$ and $\dot{s}_{\text{flx}}$)
- $\mathbf{M}$ is the mass matrix
- $\mathbf{f}$ are the force and torque equations of Force Elements
- $\mathbf{c}$ are the dynamic states of Force Elements and Control Elements (s_{dyn})
- $\mathbf{s}$ are the algebraic states (s_{alg}) on position and acceleration level
- t the model time
- $\mathbf{u}$ the values of u-Vector Elements and the excitations from driven Joints and Connections on position and velocity level
- λ the Constraint forces and torques (s_λ, only applies if the model contains Constraints or the model has a closed kinematic loop with Connections)
- $\mathbf{G}$ the Jacobian matrix of the constraint conditions, $\mathbf{G}(\mathbf{p},\mathbf{c},\mathbf{s},t,\mathbf{u}) = \frac{\mathrm{d}\mathbf{g}}{\mathrm{d}\mathbf{p}}$
- $\mathbf{f}_c$ the dynamic state equations of Force Elements and Control Elements
- $\mathbf{g}$ the algebraic constraint conditions related to Constraints (only applies if the model contains them)
- $\mathbf{b}$ the algebraic constraint conditions related to algebraic states (only applies if the model contains them)

No difference is made between independent and dependent Joint and Flexible Body states (see State Dependencies and Constraint Redundancy). In Simpack, the mass matrix $\mathbf{M}(\mathbf{p})$ is shifted to the right-hand side of equation (momentum), where it appears accordingly as $\mathbf{M}^{-1}(\mathbf{p})$, if the explicit formalism is used, see Explicit and Residual Formalism.

In this formulation, the state vector is

$$\mathbf{x}(t) = \begin{pmatrix} \mathbf{p} \\ \mathbf{c} \\ \mathbf{v} \\ \lambda \\ \mathbf{s} \end{pmatrix}$$

which directly corresponds to the symbol naming in State Vector.

Equation (kinematic) is the so-called kinematic differential equation and reduces the system from second to first order because common time integration solvers used in multibody simulation are available for first-order systems of equations only (nonlinear state-space representation).

Equation (momentum) is the well-known principle of (linear) momentum according to Newton and Euler, with the Constraint forces and torques brought in as Lagrange multipliers. Equation (dynstates) covers the first-order dynamic states of Force or Control Elements. The algebraic equation (cnsts) describes the constraint conditions used by Constraints. The second algebraic equation (algebraic) describes the conditions used for algebraic states of Force Elements and Control Elements.

Preconditions for Solvability

This system of equations is solvable if the following conditions are fulfilled:

1. Positive definite mass matrix: The mass matrix $\mathbf{M}$ must be symmetric and positive definite. Thus, Bodies with zero or even negative mass or main inertia moments are not allowed.
2. Force law continuity: All force laws $\mathbf{f}(\mathbf{p},\mathbf{v},\mathbf{c},\mathbf{s},t,\mathbf{u},\lambda)$ used in Force Elements must be continuous in all their independents. This is a tough requirement and certain effort is needed to fulfil it even in strongly nonlinear cases like contact loss and regain and stick-slip friction.

   Simpack's integrators (see Available Integrators) may, depending on the tolerance settings (Tolerances Tab), just skip small discontinuities. However, it is also possible that they significantly reduce the stepsize, and the calculation time increases drastically. If the discontinuity is too large and/or very fine tolerances are used then the integrator may even abort with a fatal error.

   Simpack's Modeling Elements and the Time Integration solver provide special functionality to smooth or circumvent discontinuous force laws. Some Force Elements and Control Elements provide root functions, causing the integrator to automatically stop and restart the integration when a discontinuity is detected, see Root Functions, Root Functions and Root Functions.
3. Unique constraint conditions: The Jacobian of the constraint conditions $\mathbf{G}$ must have full rank. This means in particular that constraining conditions must neither be redundant nor conflicting, see State Dependencies and Constraint Redundancy. Redundant constraining conditions are ignored (by default) automatically by the solver. Conflicts cause an error as the constraining conditions cannot be fulfilled and need to be resolved by the user.
4. Initial constraint conditions: The initial values of $\mathbf{p}$, $\mathbf{v}$, $\mathbf{c}$, $\mathbf{s}$, $\mathbf{g}$ and λ must be consistent. This is ensured by the Assemble System solver.
5. Constraint conditions differentiability: The constraint conditions $\mathbf{g}(\mathbf{p},\mathbf{c},\mathbf{s},t,\mathbf{u})$ must be twice continuously differentiable. This is normally ensured by the Constraints themselves.
6. Correctly defined algebraic states: The algebraic state residuals in $\mathbf{b}$ must depend on their associated algebraic states s_{alg}. Simpack's library Modeling Elements ensure this automatically, but in User Routines and Control Element 175: Algebraic State Feedback the user is responsible for defining this relationship properly.

Explicit and Residual Formalism

Most integrators use the explicit form of equation (kinematic) and the following equations. Some integrators, however, can solve the equations of motion also in an implicit form, i.e. they try to ensure that the difference of left-hand and right-hand side (e.g. $\dot{\mathbf{p}} - \mathbf{T}(\mathbf{p})\,\mathbf{v}$) becomes zero. This method is also called 'residual formalism' and is often significantly faster than the explicit formalism. See also Explicit and Residual Formalism for more information.

Linearized

The linear solvers Eigenvalues, Linear System Analysis and State-Space Matrices Export need a linearized version of the equations of motion stated in General Nonlinear. See Linearization for information on how the linearization is performed.

The state equation in the state-space representation of the (now linear and time invariant, 'LTI') system describes the eigenbehavior of the system and the influence of excitations:

$$\dot{\mathbf{x}}\left(t\right)=\mathbf{A}\mathbf{x}\left(t\right)+\mathbf{B}\mathbf{u}\left(t\right) \tag{1}$$

where

- $\mathbf{x}$ the state vector, containing the positions and velocities of the independent Joint and Flexible Body states (see State Dependencies and Constraint Redundancy) as well the Force or Control Element dynamic states, see State Vector
- t the model time
- $\mathbf{A}$ the system matrix
- $\mathbf{B}$ the input matrix
- $\mathbf{u}$ the input vector, defined via u-Vector Elements

The output equation describes the outputs of the system, which the user may have arbitrarily defined via y-Outputs:

$$\mathbf{y}\left(t\right)=\mathbf{C}\mathbf{x}\left(t\right)+\mathbf{D}\mathbf{u}\left(t\right) \tag{2}$$

with the additional variables

- $\mathbf{y}$ the output vector, containing an arbitrary set of quantities to be measured, defined via y-Outputs
- $\mathbf{C}$ the output matrix
- $\mathbf{D}$ the feedthrough matrix

All four matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ and $\mathbf{D}$ are constant.

The Eigenvalues solver only needs the system matrix $\mathbf{A}$, which describes the system's eigenbehavior. The Linear System Analysis solver requires all four matrices. The State-Space Matrices Export solver exports the matrices, along with the state, input and output vector in the linearization state, for use with external software.

Approach in Finite Element Software

Introduction to Finite Element Analysis

The Finite Element Method (FEM) is one of the numerical approaches developed over the last decades for modeling the dynamic behavior of the deformable bodies/components.

The method consists of three stages:

- Preprocessing: Once the geometry of interest has been generated using a Computer Aided Design (CAD) software, the resulting computation domain is subdivided (meshed) into elements of finite dimensions (known as finite elements) connected by nodes, and the appropriate material properties, boundary conditions and loads are applied.
- Solution: The equations for the applied nodal forces are defined in terms of the unknown nodal displacements. From this, the equations of equilibrium of each element are assembled in a matrix form (i.e. stiffness · displacement = load).
- Postprocessing: The results (i.e. displacement, element stresses and strains fields, natural frequencies) are obtained.

Although the FEM approach can be easily applied in a range of geometrically complex problems, a large number of finite elements and nodes have to be considered (up to six degrees of freedom per node) in order to sufficiently describe the dynamic behavior of a multibody system. Often the FEM is based on models with more than one million degrees of freedom, which implies high computational costs. Thus, it is usually limited in applications with comparatively small number of i.e. load cases, eigenmodes or it is applied to a certain component of a system.

The major challenge remains on describing the dynamic of a complete mechanical system. The coupling of multibody and finite element simulations has proven to be a crucial approach for this purpose. Though, in order to enable the coupling procedure during the modeling workflow, a method for reducing the size of the finite element model (degrees of freedom) is necessary (see Basics of Craigh Bampton Reduction).

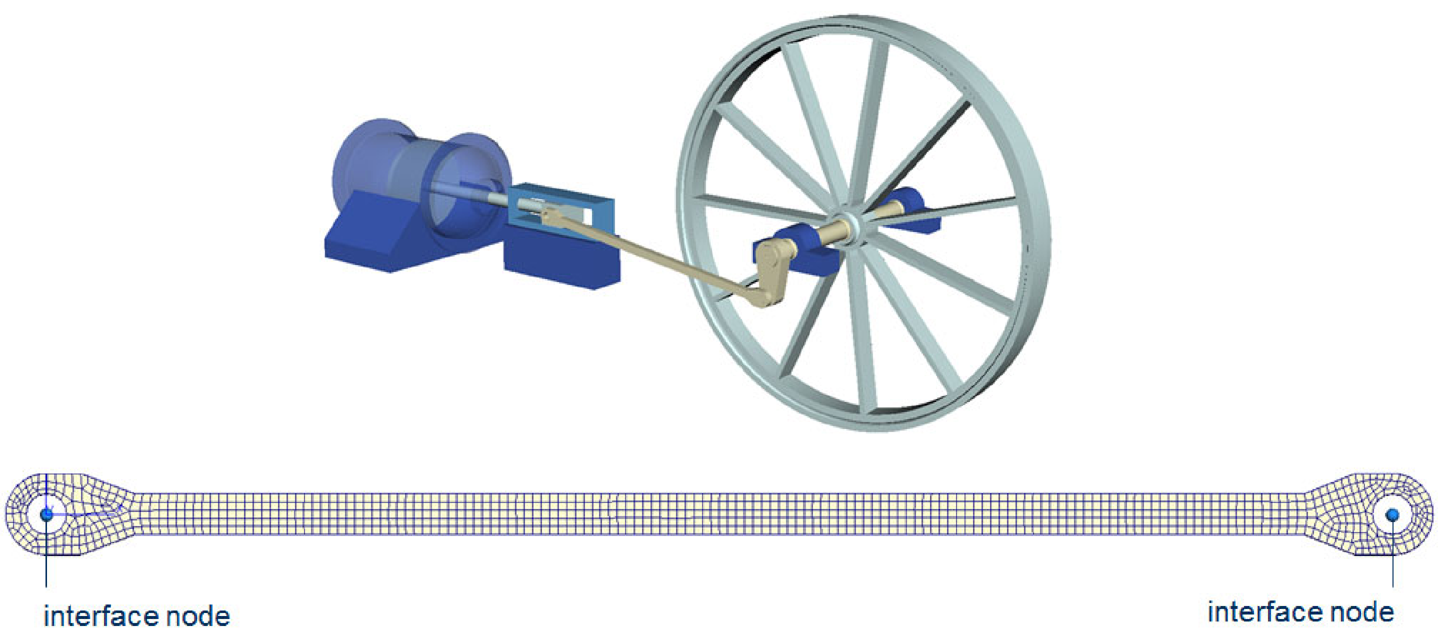
Basics of Craigh Bampton Reduction

The 'Craig-Bampton' method [Craigh1968a] is a method of reducing the size (degrees of freedom) of the FE model in order to integrate the flexible model (known as superelement) into the MBS software. The dynamic analysis is based on fundamental frequencies and the corresponding mode shapes are better performed considering a few numbers of degrees of freedom. The modes (constraint modes/constrained normal modes) of the superelement need to be specified by the user in the finite element code.

The 'Craig-Bampton method' is based on the projection of large finite element models into small matrices that contain mass, stiffness and mode shape data of the structure representing the low frequency response modes. For yielding the mode shape information, the modes are expressed in physical coordinates at the interface nodes in addition to a set of modes expressed in modal or generalized coordinates. The mode shapes corresponding to higher frequency responses, when transformed to modal coordinates, can be truncated without loss of information. The resulting matrices can be easily used for the coupling with the multibody systems.

Here, a study case is utilized for the demonstration of the introduced concepts. Figure 1 shows a multibody system of a steam mechanism with a flexible connection rod. In this example, the connection rod, reproduced in finite element code is coupled with the multibody system at the interface nodes, shown in Figure 1. In Simpack at least the interface nodes must be accessible since Simpack interacts with the FE superelement on these nodes.

Figure 1. Multibody system of a steam engine.



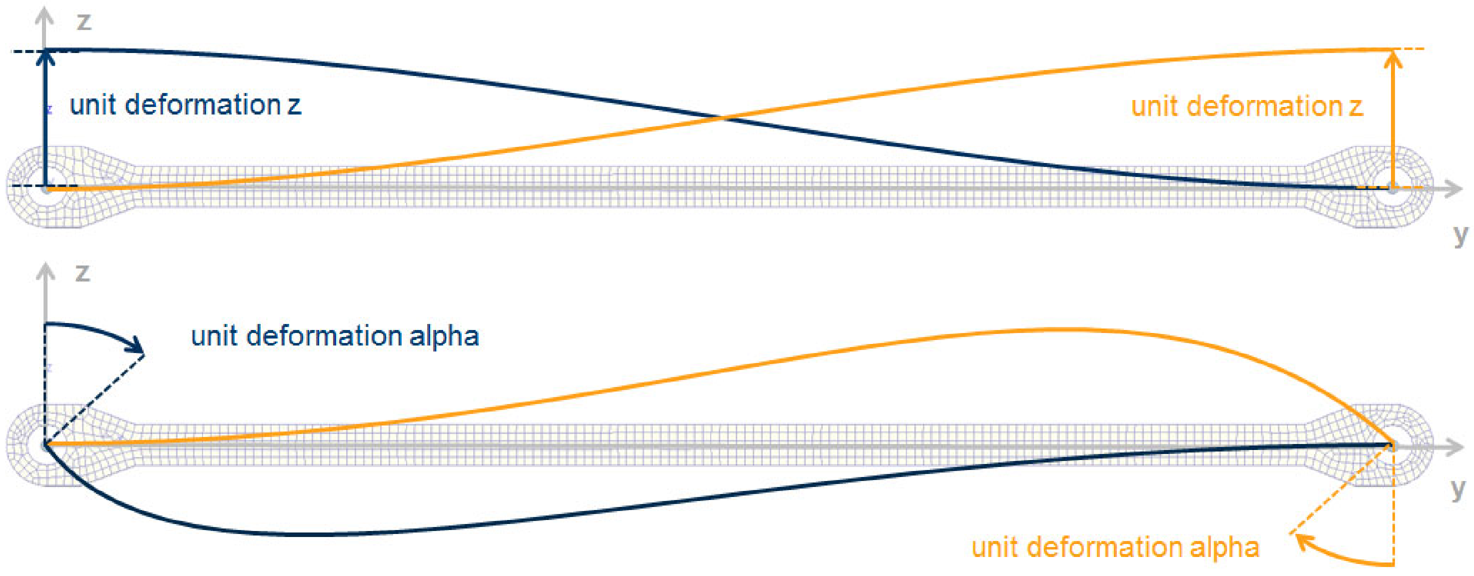
The modes that can be generated, are shown in the following sections.

Constraint Modes

The constraint modes are used to account static deformations when coupling the Body at nodes to other components through Joints, Constraints and Force Elements. These modes are the static shape assumed by the Body when setting one degree of freedom to unity while setting all other degrees of freedom at the interface nodes fixed. Consequently, the number of resulting modes is equal to the number of degrees of freedom set at the interface nodes. Accordingly, a constraint mode is computed by the finite element code for each retained degree of freedom.

In Figure 2 are shown the mode shapes obtained when setting one degree of freedom to unity at the interface nodes (retained nodes) while all others are held fixed. These constraint modes represent vertical bending. Setting one degree of freedom in the left hand side node (translation in z or rotation in a) yields the corresponding shapes coloured in blue.

Figure 2. Constraint modes representing vertical bending.



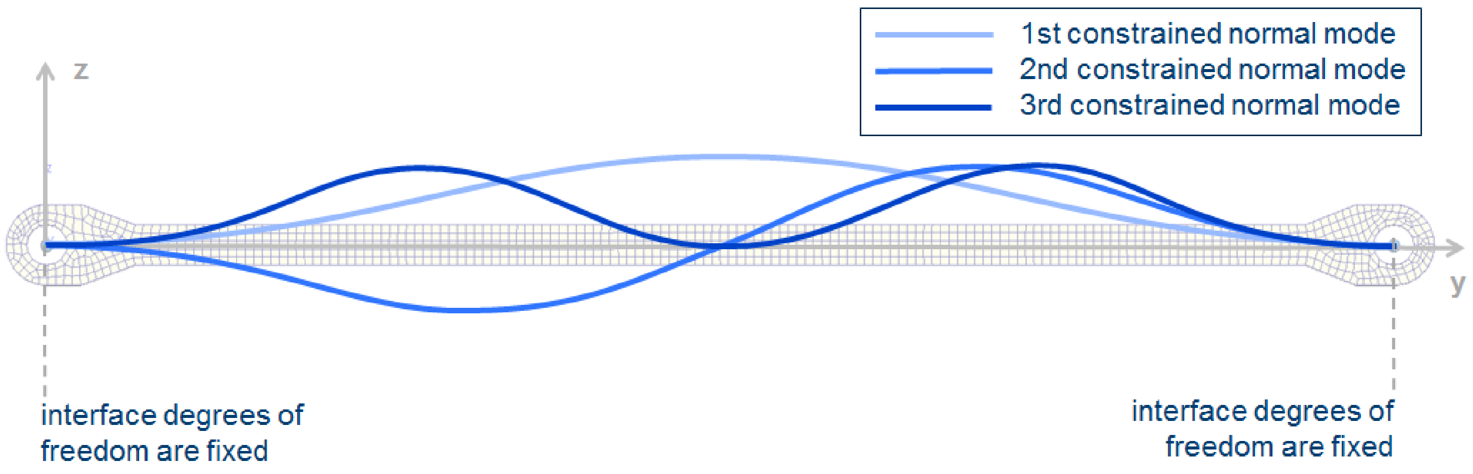
Note: The total static deformation due to forces and constraints at the interface nodes of the superelement is the linear combination of the constraint modes. The deformations at the retained nodes are the linear factors. The linear combination of constraint modes yields a statically correct solution, if concentrated loads act at the interface nodes.

Constrained Normal Modes

In order to obtain a better aproximated solution in the higher frequency range, the constrained normal modes can be computed. Constrained normal modes are the eigenmodes when all interface degrees of freedom are held fixed representing the natural vibration of the Body. The finite element code computes these modes by performing a modal analysis whilst all degrees of freedom are constrained at the interface nodes. The more constrained normal modes are considered, the better deformation results are obtained in Simpack.

In Figure 3 are shown representative mode shapes obtained when setting zero degrees of freedom at the interface nodes.

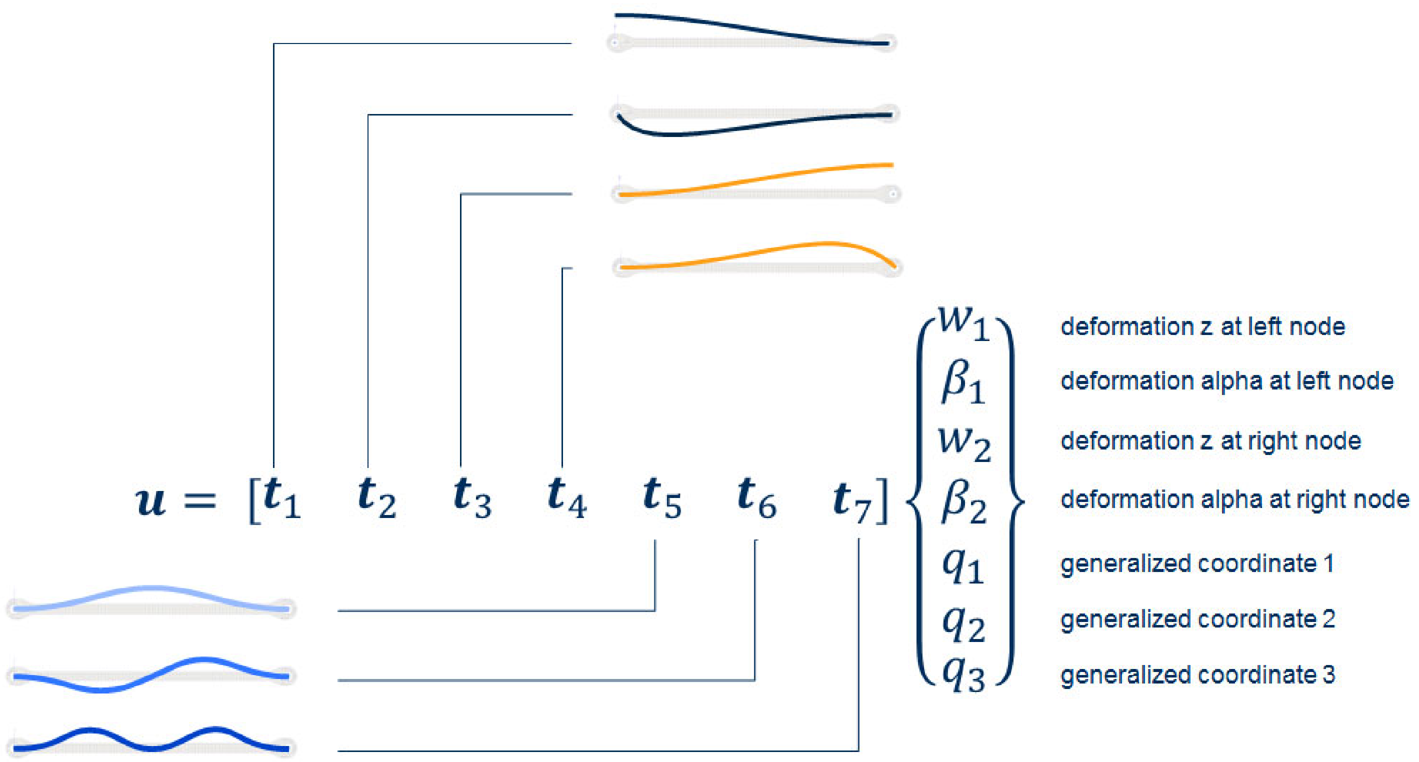
Figure 3. Total static deformation of the superelement.


1st constrained normal mode
2nd constrained normal mode
3rd constrained normal mode
z
y
interface degrees of freedom are fixed
interface degrees of freedom are fixed

Reduced Finite Element Model

The total deformation of the super element is the linear combination of constraint modes and constrained normal modes. The coordinates of the super elements are the retained degrees of freedom (deformation of interface nodes) and the generalised coordinates of the constrained normal modes. The total deformation can be written as follows:

Figure 4. Total Deformation.



$u = [t_1 \; t_2 \; t_3 \; t_4 \; t_5 \; t_6 \; t_7]$
w_1 deformation z at left node
β_1 deformation alpha at left node
w_2 deformation z at right node
β_2 deformation alpha at right node
q_1 generalized coordinate 1
q_2 generalized coordinate 2
q_3 generalized coordinate 3

The recovery matrix $\mathbf{T}$ collects all the mode shapes. The expanded solution of the reduced model deformation can be written as follows:

$$\mathbf{u} = [\mathbf{t}_1 \; \mathbf{t}_2 \; \mathbf{t}_3 \; \mathbf{t}_4 \; \mathbf{t}_5 \; \mathbf{t}_6 \; \mathbf{t}_7] \begin{Bmatrix} \mathbf{w}_1 \\ \beta_1 \\ \mathbf{w}_2 \\ \beta_2 \\ \mathbf{q}_1 \\ \mathbf{q}_2 \\ \mathbf{q}_3 \end{Bmatrix}$$

$$\mathbf{u} = \mathbf{T}\mathbf{u}_{\mathbf{SE}}$$

where, $\mathbf{T}$ is the recovery matrix. $\mathbf{u}_{\mathbf{SE}}$ is the vector of the coordinates of the reduced finite element model

The equations of motion for the finite element model before the reduction can be written as:

$$\mathbf{M}_{\mathbf{FEM}}\ddot{\mathbf{u}} + \mathbf{D}_{\mathbf{FEM}}\dot{\mathbf{u}} + \mathbf{K}_{\mathbf{FEM}}\mathbf{u} = \mathbf{p}_{\mathbf{FEM}}$$

where, $\mathbf{u}$ is the deformation vector which contains the nodal displacements. $\ddot{\mathbf{u}}$ is the second derivative value with respect to time.

The matrices and vectors (mass matrix $\mathbf{M}_{\mathbf{FEM}}$, damping matrix $\mathbf{D}_{\mathbf{FEM}}$, stiffness matrix $\mathbf{K}_{\mathbf{FEM}}$, load vectors $\mathbf{p}_{\mathbf{FEM}}$) of the finite element model are reduced using the recovery matrix $\mathbf{T}$ as shown below to yield the reduced finite element model:

$$\mathbf{M}_{\mathbf{SE}} = \mathbf{T}^{\mathbf{T}}\mathbf{M}_{\mathbf{FEM}}\mathbf{T}$$

$$\mathbf{D}_{\mathbf{SE}} = \mathbf{T}^{\mathbf{T}}\mathbf{D}_{\mathbf{FEM}}\mathbf{T}$$

$$\mathbf{K}_{\mathbf{SE}} = \mathbf{T}^{\mathbf{T}}\mathbf{K}_{\mathbf{FEM}}\mathbf{T}$$

$$\mathbf{p}_{\mathbf{SE}} = \mathbf{T}^{\mathbf{T}}\mathbf{p}_{\mathbf{FEM}}$$

where, $\mathbf{T}^{\mathbf{T}}$ is the transpose matrix of $\mathbf{T}$.

The equations of motion for the reduced finite element model can be written as:

$$\mathbf{M}_{\mathbf{SE}}\ddot{\mathbf{u}}_{SE} + \mathbf{D}_{\mathbf{SE}}\dot{\mathbf{u}}_{SE} + \mathbf{K}_{\mathbf{SE}}\mathbf{u}_{SE} = \mathbf{p}_{\mathbf{SE}}$$

The superelement matrices ($\mathbf{M}_{\mathbf{SE}}$, $\mathbf{D}_{\mathbf{SE}}$, $\mathbf{K}_{\mathbf{SE}}$) and vectors ($\mathbf{p}_{\mathbf{SE}}$) of the reduced finite element model are used for the modal Approach in Simpack.

Approach in Simpack

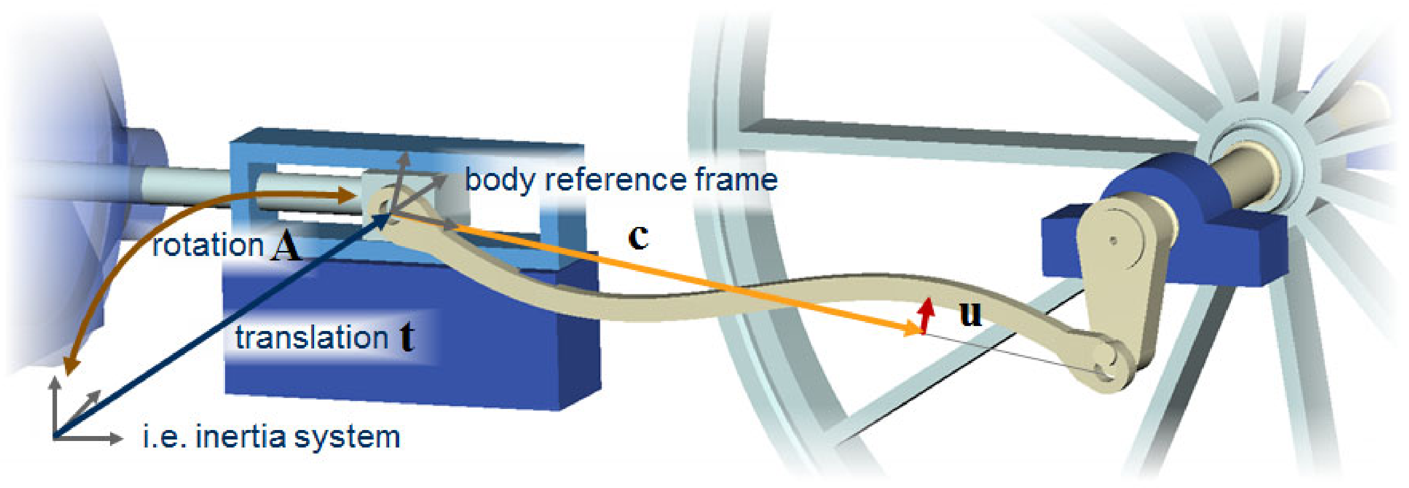
This section briefly describes by means of equations, how Simpack approaches the behavior of Flexible Bodies that are integrated into multibody systems. The Flexible Bodies are connected at the interface points by Force Elements, Constraints and Joints with the multibody system. When loads act in these connections the Flexible Bodies undergo deformations, which is represented by mode shapes. These mode shapes are either eigenmodes or modes that account for the local deformation at the interface points. The next sections will explain the equations of motion that are generated by Simpack.

Kinematic Description of Flexible Bodies in Simpack

The position of a point of the Flexible Body (kinematic description) is input for setting up the equations of motion using the principle of virtual work as will be explained in Equations of Motion. For this, before describing the equations of motion in Equations of Motion, a brief description is given here by using an example of a multibody system, on how this position is represented.

Figure Figure 1 shows the multibody system of a steam mechanism with a flexible connection rod.

Figure 1. Multibody system of a steam engine.



The position of a point of the Flexible Body (flexible connection rod):

$$\mathbf{r}(\mathbf{c}, \boldsymbol{t}) = \mathbf{A}(\mathbf{t} + \mathbf{c} + \mathbf{u}(\mathbf{c}, \boldsymbol{t})) \qquad (1)$$

is represented by $\mathbf{u}$, $\mathbf{c}$, $\mathbf{t}$, $\mathbf{A}$ as shown in Figure 1

where, $\mathbf{u}$ is the vector of deformation which depends on the location $\mathbf{c}$ and the time $\boldsymbol{t}$. $\mathbf{c}$ is the vector of rigid body configuration with respect to the Body Reference Frame $\mathbf{t}$ is the vector of translation of the Body Reference Frame with respect to Body Reference Frame $\mathbf{A}$ is the transformation matrix of the Body Reference Frame with respect to the inertia system.

The displacement vector $\mathbf{u}$ depends on the location $\mathbf{c}$ and the time $\mathbf{t}$ for transient load cases.

Assumption: Small, linear rigid body deformation is imposed on large rigid body motion.

Modal Approach

Mode shapes $\mathbf{\Phi_i}(\mathbf{c})$ are eigenmodes and/or interface modes and are formed by the superelemet matrices and vectors of the finite element reduced model described in Reduced Finite Element Model.

Eigenmodes The eigenmodes account for the global deformation and/or free oscillations of the structure.

$$\left(\mathbf{K_{SE}} - \omega^2 \mathbf{M_{SE}}\right) \mathbf{\Phi_i}(c) = 0$$

Interface modes The interface modes represent the deformation due to loads acting in Force Elements, Constraints, Joints that are connected to the Body, i.e. for the 'frequency responce mode' FRM it can be written:

$$\left(\mathbf{K_{SE}} - {\Omega_0}^2 \mathbf{M_{SE}}\right) \mathbf{\Phi_i}(c) = \mathbf{p}_{SE}$$

In Equation 1, small elastic displacements will be taken into account, expressed by linear combinations of mode shapes $\mathbf{\Phi_i}(c)$ weighted with time dependent modal coordinates $s_{\text{flx},i}(t)$, as shown in Equation 2 (modal approach):

$$\mathbf{u}(\mathbf{c}, \mathbf{t}) = \sum \mathbf{\Phi_i}(c)\, s_{\text{flx},i}(t) \qquad (2)$$

where $\mathbf{\Phi_i}(c)$ the mode shapes (eigenmodes, interface modes) $s_{\text{flx},i}(t)$ the modal coordinates

For selecting the mode shapes in the user interface, see Linear flexible Bodies: Modes Tab.

Equations of Motion

Inserting the kinematic description (position of a point of the Flexible Body $\mathbf{r}$, (see Kinematic Description of Flexible Bodies in Simpack) into the principle of virtual work, yields the equation of motion for a Flexible Body of a multibody system in which all terms of the principle of virtual work can be expressed as functions of the acceleration $\mathbf{a}$, the angular velocity ω and angular accelerating $\dot{\omega}$ and finally the modal coordinates $s_{\text{flx},i}(t)$.

Particularly, from the mode shapes $\mathbf{\Phi_i}(c)$ in Equation 3 the strain vector ε and stress vector σ are computed (Equation 4). The time dependent quantities (i.e. the modal coordinates $s_{\text{flx},i}(t)$, the transformation matrix $\mathbf{A}$) are extracted, yielding the invariants, and from the invariants the coefficients (i.e. coeefficients of the mass matrix, the corresponding coefficients of the centrifugal forces) of the equation of motion are obtained.

Position:

$$\mathbf{r^I}\left(\mathbf{c},\mathbf{t}\right)=\mathbf{A}\left(\mathbf{t}+\mathbf{c}+\sum\mathbf{\Phi_i}\left(c\right)s_{\mathrm{flx},i}\left(t\right)\right) \tag{3}$$

Principle of virtual work:

$$\int_V \rho\delta\mathbf{r}^{IT}\ddot{\mathbf{r}}^I\,\mathrm{d}V+\int_V\delta\varepsilon^T\sigma\,\mathrm{d}V=\int_V\delta\mathbf{r}^{IT}\mathbf{p}\,\mathrm{d}V \tag{4}$$

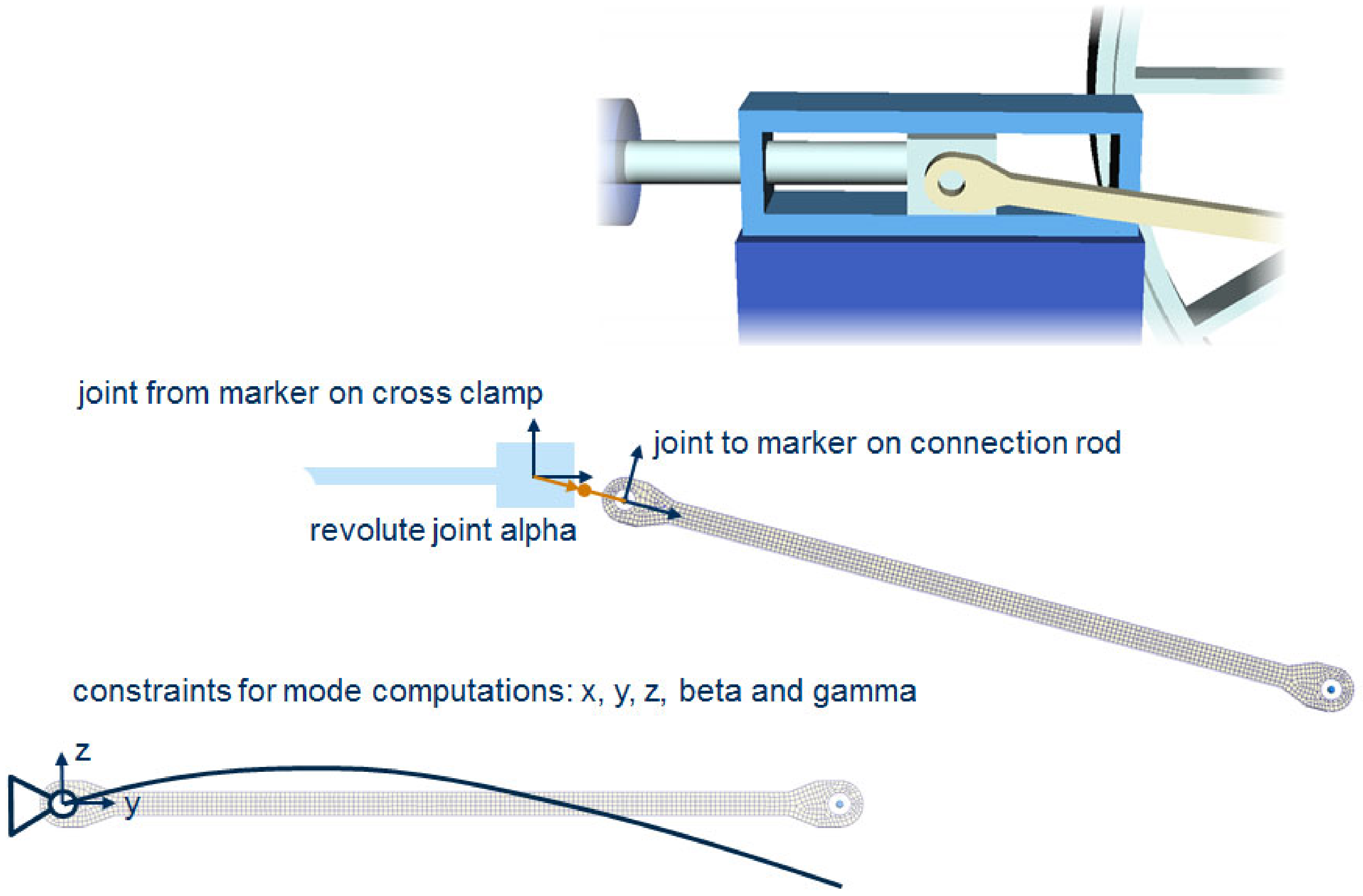
The resulting equation of motions is separated into quantities that only depend on the position (invariants over volume integral) and quantities that depend on the time. The invariants are computed by Simpack, and the computation is done automatically during the preprocessing steps (see Automatic Set-up/Update of the Equations of Motion).

Boundary Conditions for Mode Computation

The Joint of the Body determines the boundary condition that is to be considered when computing the eigenmodes and the interface modes (see Modal Approach). Directions constrained by the Joint of the Flexible Body are automatically constrained by Simpack in the mode computation steps (see Automatic Set-up/Update of the Equations of Motion).

In Figure 2 are shown the boundary conditions applied on the multibody system of a steam engine.

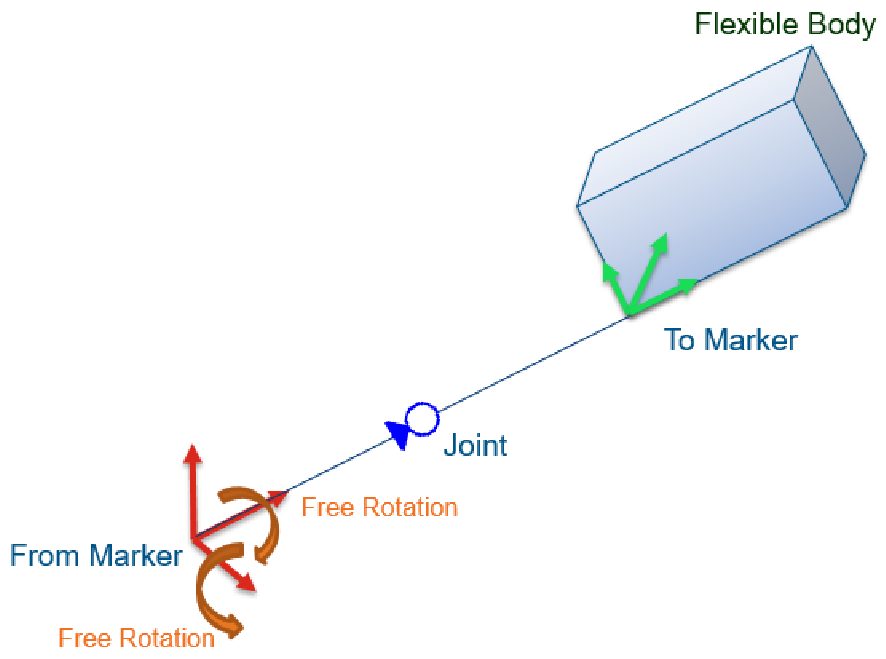
Figure 2. Boundary conditions applied on the multibody system of a steam engine.


joint from marker on cross clamp
joint to marker on connection rod
revolute joint alpha
constraints for mode computations: x, y, z, beta and gamma
z
y

Boundary Condition Requirements:

- For six degree of freedom Joints
  - Any Marker is allowed as the Joint To Marker.
- For other Joints, all of the following conditions must be met:
  - The Joint To Marker must be connected to and coincident with a single node. Please be aware that not all options of **Flexible type:** in the 'Marker Properties' dialog can be used.
  - If the Joint does not have a degree of freedom in a certain direction then the node in the FE model must have this degree of freedom (this is a requirement for the correct calculation of the Joint forces).
  - The Marker **Flexible type:** must be At node, Interpolation (Auto), Rigid Link (User) or Rigid Link (Auto).
- The boundary condition must be independent with respect to time. The Joint types that can be used must not have exactly two rotation axes. The Joint types that cannot be used are, for example, the 'universal' Joints (12: Universal al-be, 13: Universal be-ga, 14: Universal al-ga), and the Joint 25: User Defined with 2 rotational degrees of freedom.

  The free and locked directions are defined in the Joint From Marker. If two free rotations are defined in the Joint From Marker, no two rotations can be found that are permanently free in the Joint To Marker (that is, the boundary condition is time-dependent).



- When working with Connections, there are no restrictions.

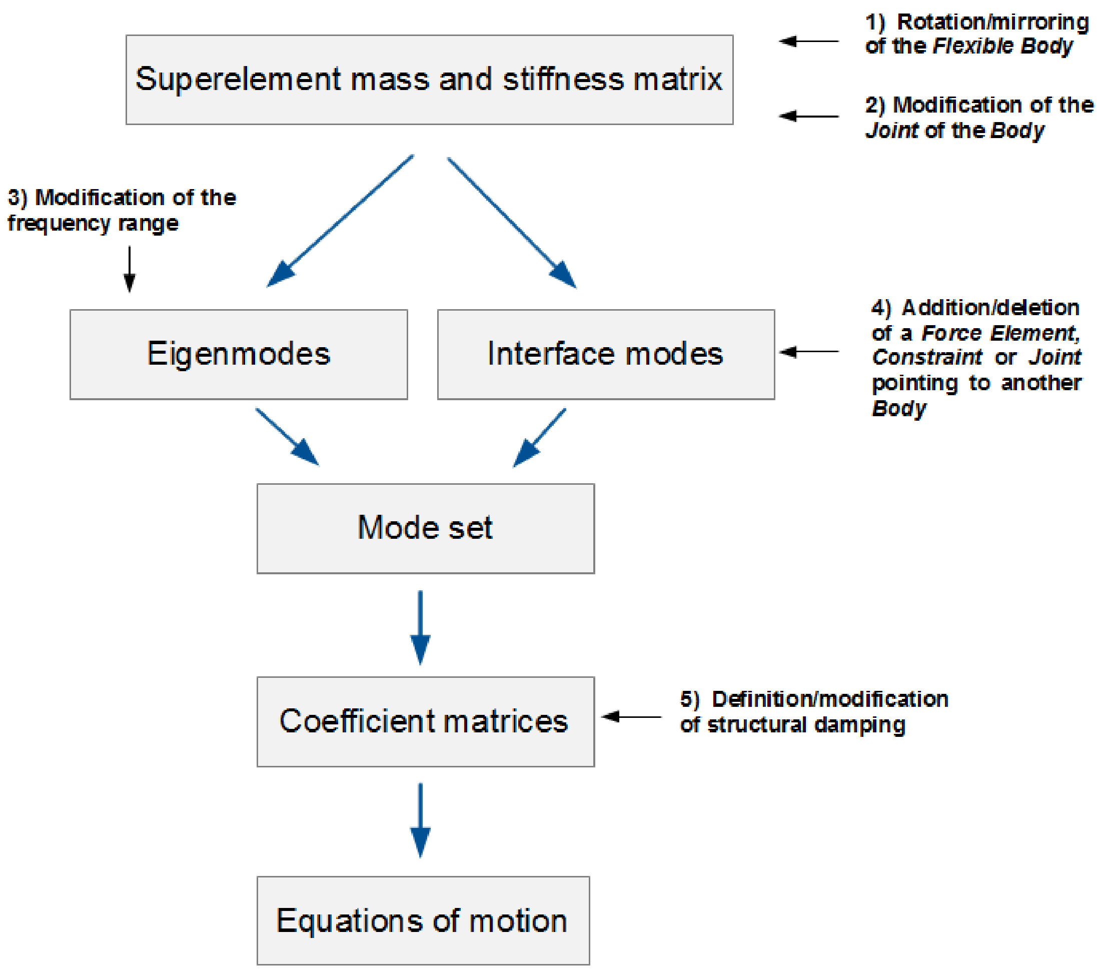
  Note: If the Joint conditions above are fulfilled, the formalism will treat the connection as a Joint. In other cases, the formalism treats the Connection as a 6 degree-of-freedom Joint with a Constraint blocking the respective motions. To check how the formalism treats the Connections, run a Test Call. Any Connections that cannot be treated as Joint are provided in the Test Call output.

Automatic Set-up/Update of the Equations of Motion

The automatic set-up/update of the equations of motion during the preprocessing steps are shown in this section.

A schematic of the automated process is illustrated in Figure 3

Figure 3. Automatic Set-up/Update of the Equations of Motion.


Superelement mass and stiffness matrix
1) Rotation/mirroring of the *Flexible Body*
2) Modification of the *Joint* of the *Body*
3) Modification of the frequency range
Eigenmodes
Interface modes
4) Addition/deletion of a *Force Element*, *Constraint* or *Joint* pointing to another *Body*
Mode set
Coefficient matrices
5) Definition/modification of structural damping
Equations of motion

The user does not intervene in the above described automated process.

Though, when working with large models, the user should go through these steps in numerical order in order to reduce the time required for setting-up the model:

1. Rotation/mirroring of the Flexible Body (see Linear flexible Bodies: FE Properties Tab)

2. Modification of the Joint type of the Body (see Joints)
3. Modification of the frequency range (see Linear flexible Bodies: Modes Tab)
4. Addition/deletion of a Force Element (see Force Elements), Constraint (see Constraints) or Joint pointing to another Body
5. Definition/modification of structural damping (see 'Damping type' option in Linear flexible Bodies: Modes Tab) and 'Damping matrix' option in Linear flexible Bodies: Options Tab.

Linear SIMBEAM Bodies

This section describes the modeling assumptions when using Linear SIMBEAM Bodies.

All SIMBEAM element formulations have 3 translational and 3 rotational degrees of freedom per node. Hence the element matrices have 12 degrees of freedom per element.

Shape Functions and Stress Resultants

This section describes the local element frame of SIMBEAM element formulations. The element's axis coincides with the x-axis of the element reference frame. The element y-axis points into the 'width-direction' of a basic Cross Section (for example; 1: Circle, 4: Rectangle), whereas the element z-axis points into the according 'height-direction'. All SIMBEAM element formulations representing linear flexible behavior are using the set of shape functions shown in the following table:

Direction	**Shape Function**	**Degrees of freedom per node**
Longitudinal deformation	Linear Lagrange polynomial	Translational deformation in the element x-direction
Torsion	Linear Lagrange polynomial	Rotational deformation about element x-direction
Bending	Cubic Hermite polynomial	Translational deformation in element y/z-direction and rotational deformation about the element z/y direction

Hence, the stress resultants have the following form along the element axis:

Stress resultant	**Form along the element axis**
Longitudinal force	Constant (discontinuous at nodes)
Torsional moment	Constant (discontinuous at nodes)
Bending moment	Linear (C0 steady)
Bending/Shear force	Constant (discontinuous at nodes)

Kinematic Assumptions

Linear SIMBEAM element formulations use either the Euler-Bernoulli Hypothesis or the Timoshenko hypothesis. In SIMBEAM's linear elements, warping is not yet taken into account, that is, the cross-sections are supposed to be 'rigid' during the deformation so that the deformation is only represented by the beam axis. In addition, the kinematic behavior, as described in the following table, is associated to the Cross Section .

Cross Section	**Comment**
4: Rectangle, 5: Rectangle, hollow	Longitudinal and bending deformation uncoupled, torsion and bending uncoupled, the two bending directions are uncoupled
1: Circle, 2: Circle, hollow	Longitudinal and bending deformation uncoupled, torsion and bending uncoupled, the two bending directions are uncoupled.
6: Ellipse, 7: Ellipse, hollow	Longitudinal and bending deformation uncoupled, torsion and bending uncoupled, the two bending directions are uncoupled.
10: General Basic	Longitudinal and bending deformation uncoupled, torsion and bending uncoupled, the two bending directions are uncoupled. 10: General Basic Cross Sections can be used for modeling arbitrarily shaped double symmetric cross-sections.
11: General Advanced	Takes into account twist bend coupling and coupling of bending and longitudinal deformation. 11: General Advanced Cross Sections can be used for modeling arbitrarily shaped cross-sections. In addition, depending on the cross-section's shape, the two bending directions may be coupled. The latter effect is described by the area deviation moment of inertia, see Table 1.
12: Matrix Input	Not yet available for linear SIMBEAM element formulations. Composite Beam technology with in-plane and out-plane warping. Fully populated 6x6 beam material matrix, which is calculated from 2D or 3D cross-sectional FE representations. This allows you to use detailed stress recovery in prismatic beam sections. The warping is always free, that is, also this formulation has 12 degrees of freedom per element.

Input

Table 1 shows the data that is used to the setup the element formulations representing linear elastic behavior:

Table 1. Input Data

Input to Element Matrices	**Symbol used in formulae**	**Comment**
Young's Modulus	E	It is used to model isotropic linear material.
Shear Modulus	G	It is used to isotropic linear material.
Cross-section area	A	Bending stiffness when multiplied with Young's modulus. Input to mass matrix when multiplied with density.
Area moment of inertia about element x-axis	I_x	Input to element mass matrix.
Area moment of inertia about element y-axis	I_y	Bending stiffness when multiplied with Young's modulus. Input to mass matrix when multiplied with density.

Area moment of inertia about element z-axis	I_z	Bending stiffness when multiplied with Young’s modulus. Input to mass matrix when multiplied with density.
Torsional stiffness constant	I_t	Yields torsional stiffness when multiplied with shear modulus.
Area deviation moment of inertia	I_{yz}	For 11: General Advanced Cross Section only.
Static moment about element y-axis	S_y	For 11: General Advanced Cross Section only.
Static moment about element z-axis	S_z	For 11: General Advanced Cross Section only.
Shear center offset in element y-direction	y_{CS}	For 11: General Advanced Cross Section only.
Shear center offset in element z-direction	z_{CS}	For 11: General Advanced Cross Section only.
Center of gravity offset in element y-direction	y_{CG}	For 11: General Advanced Cross Section only.
Center of gravity offset in element z-direction	z_{CG}	For 11: General Advanced Cross Section only.
Shear factor in element y-direction	ψ_y	The dimensionless shear factor describes the amount of shearing in a range from 0 to 1. With a shear factor of 0, the SIMBEAM elements to which the Cross Section is assigned generate the maximum of shear deformation, whereas no shear deformation occurs if the shear factor is 1.
Shear factor in element z-direction	ψ_z	The dimensionless shear factor describes the amount of shearing in a range from 0 to 1. With a shear factor of 0, the SIMBEAM elements to which the Cross Section is assigned generate the maximum of shear deformation, whereas no shear deformation occurs if the shear factor is 1.

The shear factors are given by:

$$\psi_z = \frac{1}{1+12\frac{EI_Y}{A_{sz}}} \tag{1}$$

$$\psi_y = \frac{1}{1+12\frac{EI_z}{A_{sy}}}$$

where the shear area A_s is defined by:

$$\bar{\tau} A_S = \int \tau \, dA$$

with the averaged (that is, constant) shear stress $\bar{\tau}$.

Element Matrices

The element length is given by l. The rest of the symbols are explained in Table 1.

Stiffness

Longitudinal Deformation

Stiffness

$$K_l = \frac{EA}{l}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

Mass

$$M_l = \rho A l \begin{pmatrix} 1/3 & 1/6 \\ 1/6 & 1/3 \end{pmatrix}$$

Torsional Deformation

Stiffness

$$K_t = \frac{GI_t}{l}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

Mass

$$M_t = I_x l \begin{pmatrix} 1/3 & 1/6 \\ 1/6 & 1/3 \end{pmatrix}$$

Bending

Stiffness for bending in z-direction

$$K_{bz} = \frac{EI_y}{l^3}\begin{pmatrix} 12\psi & -12\psi & -6l\psi & -6l\psi \\ -12\psi & 12\psi & 6l\psi & 6l\psi \\ -6l\psi & 6l\psi & l^2\left(1+3\psi\right) & l^2\left(-1+3\psi\right) \\ -6l\psi & 6l\psi & l^2\left(-1+3\psi\right) & l^2\left(1+3\psi\right) \end{pmatrix}$$

Mass for bending in z-direction

$$M_{bz} = \frac{\rho Al}{840}\begin{pmatrix} 4\left(70+7\psi+\psi^2\right) & 4\left(35-7\psi-\psi^2\right) & -l\left(35+7\psi+2\psi^2\right) & l\left(35-7\psi-2\psi^2\right) \\ 4\left(35-7\psi-\psi^2\right) & 4\left(70+7\psi+\psi^2\right) & -l\left(35-7\psi-2\psi^2\right) & l\left(35+7\psi+2\psi^2\right) \\ -l\left(35+7\psi+2\psi^2\right) & -l\left(35-7\psi-2\psi^2\right) & l^2\left(7+\psi^2\right) & -l^2\left(7-\psi^2\right) \\ l\left(35-7\psi-2\psi^2\right) & l\left(35+7\psi+2\psi^2\right) & -l^2\left(7-\psi^2\right) & l^2\left(7+\psi^2\right) \end{pmatrix}$$

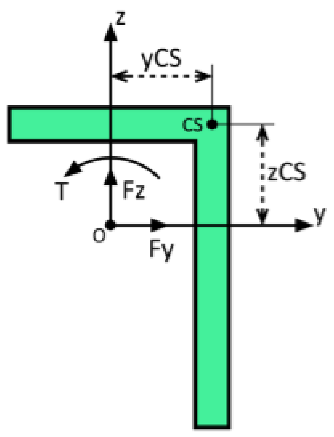
In the above listed bending matrices translational deformation is described by the rows 1 and 2, and rotational deformation is described by the rows 3 and 4. Setting A_s to 1 in the above bending matrices yields the Euler-Bernoulli element formulation. For the Timoshenko beam formulation, the shear factor is $0 < \psi < 1$.

Influence of the Shear Center Offsets

With the 11: General Advanced Cross Section, you can take into account coupling terms between torsion and bending

Figure 1 shows the coordinate system of the cross-section with origin o and the axes y and z. The beam axis is perpendicular to the drawing plane and located at the origin o. The shear forces F_y and F_Z are the resultants of the shear stress in y- and z-direction. The shear center has the offset y_{CS} and z_{CS} with respect to the origin o. The 11: General Advanced Cross Section takes the influence of the shear center into account as an additional torsional moment T that follows from the equilibrium $T = F_y z_{CS} - F_z y_{CS}$. T is imposed on the torsional moment.

Figure 1. Schematic of the shear center definition



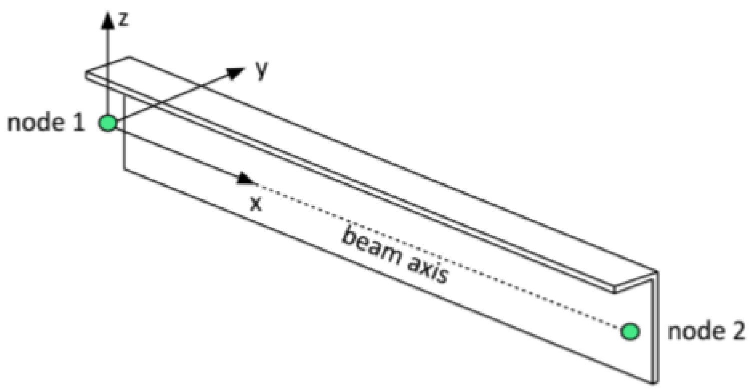
Taking this into account in the derivation of the beam stiffness matrix yields the shear center influence as shown in the following table:

Table 2. Shear center influence (expressions that couple bending in z-direction with torsion)

	Torsion at node 1	Torsion at node 2
Translational bending deformation in z-direction at node 1	$\frac{12E\psi(I_y y_{CS}+I_{yz} z_{CS})}{l^3}$	$-\frac{12E\psi(I_y y_{CS}+I_{yz} z_{CS})}{l^3}$
Translational bending deformation in z-direction at node 2	$-\frac{12E\psi(I_y y_{CS}+I_{yz} z_{CS})}{l^3}$	$\frac{12E\psi(I_y y_{CS}+I_{yz} z_{CS})}{l^3}$
Rotational bending deformation about y-direction at node 1	$-\frac{6E\psi(I_y y_{CS}+I_{yz} z_{CS})}{l^2}$	$-\frac{6E\psi(I_y y_{CS}+I_{yz} z_{CS})}{l^2}$
Rotational bending deformation about y-direction at node 2	$-\frac{6E\psi(I_y y_{CS}+I_{yz} z_{CS})}{l^2}$	$-\frac{6E\psi(I_y y_{CS}+I_{yz} z_{CS})}{l^2}$

Bending in y-direction is modeled accordingly. Figure 2 shows the element reference system, the position of node 1, and the position of node 2.

Figure 2. Beam coordinate system and nodes



Damping

Linear SIMBEAM element formulations assume viscous damping directly in the modal space, therefore, no damping matrices are used. The modal damping matrix is supposed to be diagonal.

Rotary Inertia

In the SIMBEAM mass matrix formulation, the moments of inertia are taken into account $\theta_y = \int z^2 \mathbb{d}m$, which is often not the case in standard beam element formulations. The above formula describes the rotary inertia about the element y-axis and the coordinate z is also defined in the element reference frame.
The following example of a shaft with a circular cross-section shows the effects of the rotary inertia terms:
Natural frequencies in 1/s, ratio length/diameter=5, kinematics: Euler-Bernoulli

Bending mode	**SIMBEAM without rotary inertia**	**SIMBEAM with rotary inertia**	**ANSYS (volume elements)**	**Reference from Gasch1989a**
1	1151	1053	979	1163
2	3173	2655	2098	3208

Natural frequencies in 1/s, ratio length/diameter=5, kinematics: Euler-Bernoulli

Bending mode	**SIMBEAM without rotary inertia**	**SIMBEAM with rotary inertia**	**ANSYS (volume elements)**	**Reference from Gasch1989a**
1	288	276	274	290
2	793	711	696	802

As expected, the influence of the rotary inertia terms increases with the ratio diameter/length and the eigenfrequency. The rotary inertia terms shift the eigenfrequencies toward a finite element representation. You cannot deactivate the rotary inertia terms.