

**PUBLIC**

Document Version: Release 2211

# User's Guide

Enterprise Product Development  
Connected Products

**Fedem 7.6**

---

Copyright © 2019-2022 by SAP SE.

Copyright © 1995-2018 by Fedem Technology AS.

Published 2022.

All rights reserved. No part of this document may be reproduced in any form or distributed in any way without prior written permission from SAP SE.

The information provided in this document is subject to change without notice.

The Software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement, which accompanies the software. The Software uses the following third-party software modules, each having their own copyright/license terms, as follows:

Qt : Covered by the GNU Lesser General Public License (LGPL).

Coin : Copyright © Kongsberg Oil & Gas Technologies AS.

SoQt : Copyright © Kongsberg Oil & Gas Technologies AS.

Simage : Copyright © Kongsberg Oil & Gas Technologies AS.

SmallChange : Copyright © Kongsberg Oil & Gas Technologies AS.

Qwt : The curve plotting is based on part of the work of the Qwt project (<http://qwt.sf.net>).

zlib; Copyright © Jean-loup Gailly and Mark Adler.

FMI; Copyright © Modelica Association Project "FMI".

AeroDyn : Copyright © National Renewable Energy Laboratory

TurbSim : Copyright © National Renewable Energy Laboratory

The full license text of these modules is reproduced in the file LICENSES.txt, which is included with the Software installation.

**SAP Norway Engineering Center of Excellence**

Nedre Bakkestrand 58C, N-7014 Trondheim, Norway

---

# Table of Contents

## Chapter 1 Introduction to Fedem

1.1	What is Fedem? .....	1-2
1.2	Nonlinear structural dynamics.....	1-3
1.3	Control systems in mechanical analysis .....	1-3
1.4	What is a Fedem model? .....	1-4
1.5	Using FE models .....	1-4
1.6	Fedem solver modules .....	1-5
1.6.1	FE Part Solver .....	1-5
1.6.2	Reducer .....	1-5
1.6.3	Dynamics Solver .....	1-6
1.6.4	Stress Recovery .....	1-6
1.6.5	Mode Shape Recovery .....	1-6
1.6.6	Strain Rosette Analysis .....	1-6
1.6.7	Strain Coat Analysis .....	1-6
1.6.8	Curve Export Utility .....	1-6

## Chapter 2 Learning the Basics

2.1	System requirements .....	2-2
2.2	Storing models and results .....	2-2
2.2.1	FTL format.....	2-3
2.2.2	FTC format .....	2-3
2.2.3	Other supported formats .....	2-3
2.3	Starting Fedem .....	2-3
2.3.1	Command-line options .....	2-4
2.3.2	Template model file .....	2-4

---

2.3.3	Console window .....	2-5
2.4	Touring the interface.....	2-6
2.4.1	Main window .....	2-6
2.4.2	Menus and tool bars.....	2-7
2.4.3	Model Manager .....	2-9
2.4.4	ID and Topology panel .....	2-10
2.4.5	Property Editor.....	2-11
2.4.6	Object Browser.....	2-13
2.4.7	Workspace .....	2-15
2.4.8	Output List.....	2-18
2.5	Executing commands .....	2-18
2.5.1	Select .....	2-19
2.5.2	Done .....	2-20
2.5.3	Cancel .....	2-20
2.6	Visualizing the model .....	2-21
2.6.1	3D Navigation.....	2-21
2.6.2	Predefined view tool buttons .....	2-23
2.6.3	3D View controls .....	2-24
2.6.4	Zoom and Pan .....	2-24
2.6.5	General Appearance.....	2-26
2.6.6	Item Appearance.....	2-29
2.6.7	Element face visibility .....	2-31
2.6.8	Visualization of special elements.....	2-31
2.6.9	Measuring distance and angles in a model .....	2-32
2.7	Opening and saving model files .....	2-33
2.7.1	Opening a file .....	2-33
2.7.2	Saving models .....	2-34
2.7.3	Starting a new model.....	2-36
2.8	Loading and unloading FE-Data .....	2-37

---

2.8.1	FE-Data Settings .....	2-37
2.8.2	Skipping FE-Data when opening a model file .....	2-37
2.8.3	Modeling with unloaded parts .....	2-38
2.8.4	Postprocessing unloaded parts.....	2-38
2.9	Exporting objects .....	2-38
2.9.1	Exporting a part .....	2-39
2.9.5	Exporting animations.....	2-41
2.10	License information.....	2-41
2.10.1	Available modules .....	2-41
2.10.2	License denial .....	2-42
2.10.3	License file/server .....	2-42
2.10.4	Managing license files/servers.....	2-42
2.11	Customizing Fedem using plug-ins .....	2-43
2.11.1	User-defined functions.....	2-43
2.11.2	User-defined elements.....	2-45
2.12	Customizing Fedem using Addons .....	2-49

## Chapter 3 Wind Power Modeling

3.1	Wind turbine modeling approaches .....	3-2
3.2	Creating a basic wind turbine model .....	3-3
3.2.1	Turbine definition .....	3-5
3.2.2	A newly generated wind turbine model .....	3-8
3.3	Basic solving and analysis .....	3-15
3.3.1	Running the dynamics solver (basic mode) .....	3-16
3.3.2	Basic results overview.....	3-17
3.3.3	Reliability and validity of results .....	3-18
3.4	Creating an advanced wind turbine model .....	3-19
3.4.1	Airfoil definition .....	3-20
3.4.2	Blade definition.....	3-22

---

3.4.3	Aerodynamic setup .....	3-27
3.4.4	Tower definition .....	3-33
3.4.5	Control system.....	3-34
3.5	Advanced solving and analysis .....	3-36
3.5.1	Running the dynamics solver (advanced mode) .....	3-36
3.5.2	Advanced results analysis.....	3-37

## Chapter 4 Mechanism Modeling

4.1	Basic assembling techniques .....	4-2
4.2	Mechanism modeling environment.....	4-2
4.2.1	Modeler view .....	4-3
4.2.2	Modeling tool bars .....	4-3
4.3	Mechanism modeling tools.....	4-4
4.3.1	Reference Plane .....	4-4
4.3.2	Interactive Odometer and 3D Point Marker.....	4-5
4.3.3	Stickers .....	4-6
4.4	Creating mechanism elements .....	4-7
4.4.1	Selecting position and orientation .....	4-9
4.5	Moving mechanism elements .....	4-10
4.5.1	Smart Move .....	4-10
4.5.2	Align CS and rotations.....	4-12
4.5.3	Move To Center .....	4-13
4.5.4	Origin property .....	4-14
4.6	Attaching and detaching elements .....	4-16
4.6.1	Using the <i>Attach command</i> .....	4-16
4.6.2	Surface Connectors.....	4-17
4.6.3	Surface connector commands .....	4-18
4.6.4	Attachment rules and restrictions.....	4-20
4.6.5	Detaching.....	4-21

---

4.6.6	Color of attached and unattached elements .....	4-21
4.6.7	Invalid attachments.....	4-21
4.7	Deleting mechanism elements .....	4-22
4.7.1	Deleting in the Modeler view.....	4-22
4.7.2	Deleting in the Model Manager panel .....	4-22
4.8	Using file references in mechanism elements .....	4-23
4.9	Model preferences.....	4-24
4.9.1	Model database units .....	4-25
4.9.2	Modeling tolerance.....	4-25
4.9.3	Gravitation .....	4-26
4.9.4	Initial translational velocity.....	4-26
4.9.5	External function values from file.....	4-26

## Chapter 5 Mechanism Elements

5.1	Parts .....	5-2
5.1.1	Visualization of beam element orientation and eccentricity in FE Parts...	5-4
5.1.2	Creating parts by file import .....	5-5
5.1.3	Creating parts from hard points .....	5-6
5.1.4	Copying parts .....	5-7
5.1.5	Part properties.....	5-7
5.1.6	Using FE model repositories.....	5-17
5.2	Element groups.....	5-18
5.2.1	Element group properties.....	5-19
5.3	Beams.....	5-20
5.3.1	Creating beams.....	5-20
5.3.2	Splitting beams.....	5-21
5.3.3	Beam properties.....	5-22
5.4	Beam cross sections.....	5-23
5.4.1	Material .....	5-24

---

5.4.2	Import of generic cross sections .....	5-25
5.5	Triads.....	5-26
5.5.1	Triads in joints.....	5-26
5.5.2	Triad symbols .....	5-27
5.5.3	Triad properties .....	5-28
5.6	Joints.....	5-31
5.6.1	Joint variables.....	5-32
5.6.2	Joint properties .....	5-33
5.6.3	Point-to-point joints.....	5-40
5.6.4	Point-to-path joints .....	5-44
5.7	Joint pair constraints.....	5-51
5.7.1	Gear .....	5-51
5.7.2	Rack-and-Pinion.....	5-51
5.7.3	General joint pair constraints .....	5-52
5.8	Frictions .....	5-53
5.9	Springs and Dampers .....	5-54
5.9.1	Spring properties.....	5-54
5.9.2	Damper properties .....	5-55
5.9.3	Axial spring symbol .....	5-56
5.9.4	Axial damper symbol .....	5-56
5.9.5	Spring and damper characteristics .....	5-57
5.9.6	Advanced spring characteristics .....	5-59
5.10	Loads.....	5-60
5.10.1	Load symbols .....	5-60
5.10.2	Load properties .....	5-61
5.10.3	Target point.....	5-61
5.10.4	Direction.....	5-62
5.11	Functions.....	5-62
5.11.1	Creating a function.....	5-63

---

5.11.2	Function properties.....	5-63
5.11.3	Preview .....	5-64
5.11.4	Extrapolation .....	5-65
5.11.5	Function Types .....	5-65
5.11.6	Time history input files.....	5-76
5.12	Sensors.....	5-76
5.12.1	Simple sensors.....	5-77
5.12.2	Relative sensors .....	5-77
5.12.3	Managing sensors .....	5-77
5.13	Strain rosettes .....	5-78
5.14	Generic database objects .....	5-80
5.15	Sub-assemblies .....	5-80
5.15.1	Creating sub-assemblies .....	5-81
5.15.2	Sub-assembly properties.....	5-81
5.15.3	Importing sub-assemblies.....	5-82
5.15.4	User ID convention in assemblies.....	5-83

## Chapter 6 Marine Modeling

6.1	Sea environment .....	6-2
6.1.1	Coordinate system for wave kinematics .....	6-3
6.1.2	Marine growth.....	6-4
6.2	Wave and current functions .....	6-4
6.2.1	Sea wave functions .....	6-4
4.	Sea current functions .....	6-11
6.3	Response amplitude operators .....	6-12
6.3.1	Creating RAOs .....	6-12
6.3.2	RAO table format .....	6-14
6.4	Beam string structures .....	6-15
6.4.1	Import of beam strings.....	6-15

---

6.4.2	Beam string file format .....	6-16
6.4.3	Beam string properties .....	6-18
6.4.4	Mooring line generation.....	6-19
6.5	Space frame structures .....	6-20
6.5.1	Import of space frames .....	6-20
6.5.2	Space frame properties.....	6-21
6.6	Soil Pile structures .....	6-22
6.6.1	Import of soil piles.....	6-22
6.6.2	Soil pile file format .....	6-23
6.6.3	Secant stiffness on unloading.....	6-25
6.6.4	Soil pile property .....	6-26
6.7	Hydrodynamic load calculations.....	6-27
6.7.1	Buoyancy of 3D volumes .....	6-27
6.7.2	Hydrodynamic loads on beam elements .....	6-27

## Chapter 7 Control System Modeling

7.1	Control modeling environment.....	7-2
7.1.1	Control Editor.....	7-2
7.1.2	Control tool bars .....	7-2
7.1.3	Control system topology .....	7-3
7.2	Input and output.....	7-3
7.3	Control blocks .....	7-4
7.3.1	Amplifiers.....	7-4
7.3.2	Binary-input blocks.....	7-4
7.3.3	Integrator and limited derivator blocks.....	7-5
7.3.4	Time- dependent blocks.....	7-5
7.3.5	Non-continuous blocks.....	7-5
7.3.6	PI, PD, and PID controllers .....	7-6
7.3.7	General-transfer functions .....	7-7

---

7.4	Building control modules .....	7-7
7.4.1	Setting Grid and Snap .....	7-8
7.4.2	Inserting blocks.....	7-8
7.4.3	Moving blocks.....	7-8
7.4.4	Editing block properties.....	7-9
7.4.5	Connecting blocks.....	7-9
7.4.6	Rotating blocks.....	7-10
7.4.7	Deleting blocks or connections.....	7-10
<b>Chapter 8</b>	<b>Mechanism Analysis</b>	
8.1	Overview of Fedem analyses .....	8-2
8.1.1	Model reduction.....	8-2
8.1.2	Dynamics analysis .....	8-2
8.1.3	Stress recovery .....	8-4
8.1.4	Mode shape recovery.....	8-4
8.1.5	Strain rosette recovery.....	8-5
8.1.6	Strain coat recovery.....	8-5
8.1.7	Linear analysis .....	8-5
8.2	Solver tools .....	8-5
8.2.1	Model export for external simulation .....	8-5
8.2.2	Solvers tool bar .....	8-7
8.2.3	Additional solver options .....	8-8
8.2.4	Controlling placement of temporary files .....	8-11
8.2.5	Part- and group-wise solving.....	8-11
8.2.6	Viewing the progress of long-duration analyses.....	8-12
8.3	Model reduction .....	8-13
8.3.1	Starting the model reduction .....	8-13
8.3.2	Using component modes .....	8-14
8.3.3	Using lumped mass matrix.....	8-14

---

8.3.4	Handling singularities during the model reduction .....	8-15
8.3.5	Eigenvalue analysis of the reduced parts .....	8-17
8.3.6	Visualization of eigenmode shapes from the model reduction .....	8-17
8.3.7	Reduction of applied load vectors.....	8-18
8.4	<b>Model reduction in Nastran .....</b>	<b>8-18</b>
8.4.1	Nastran DMAP .....	8-18
8.4.2	Nastran bulk data entries for CMS reduction.....	8-19
8.4.3	Recovery of parts reduced in Nastran .....	8-20
8.5	<b>Dynamics analysis.....</b>	<b>8-21</b>
8.5.1	Dynamics Solver (Basic Mode) .....	8-21
8.5.2	Dynamics Solver (Advanced Mode) .....	8-22
8.5.3	Frequency response analysis.....	8-31
8.5.4	Dynamic ramp-up of loads and prescribed motions .....	8-32
8.5.5	Result output control.....	8-33
8.5.6	Stress and strain recovery during time integration.....	8-36
8.5.7	Monitoring the most problematic DOFs during time integration .....	8-37
8.5.8	Starting the analysis.....	8-38
8.5.9	Handling singularities during the dynamics analysis .....	8-38
8.6	<b>Stress recovery analysis .....</b>	<b>8-38</b>
8.6.1	Stress recovery options.....	8-38
8.6.2	Result output control.....	8-40
8.6.3	Starting the analysis.....	8-40
8.7	<b>Mode shape recovery analysis .....</b>	<b>8-40</b>
8.7.1	Mode shape options.....	8-40
8.7.2	Starting the analysis .....	8-41
8.8	<b>Strain rosette analysis .....</b>	<b>8-42</b>
8.8.1	Strain rosette options .....	8-42
8.8.2	Starting the analysis .....	8-43
8.8.3	Result output .....	8-43

---

8.8.4	Strain rosette definition file format .....	8-45
8.9	Strain coat analysis.....	8-47
8.9.1	Generating strain coat .....	8-47
8.9.2	Strain coat analysis options .....	8-48
8.9.3	Starting the analysis .....	8-49
8.9.4	Strain coat recovery on element groups or individual parts.....	8-49
8.10	Direct linear analysis of FE Parts .....	8-49
8.10.1	Starting a direct linear analysis .....	8-50
8.11	Interaction during processing .....	8-51
8.11.1	Simultaneous viewing and processing.....	8-51
8.11.2	Stop processing .....	8-51
8.12	Deleting results.....	8-52
8.12.1	Deleting specific results.....	8-52
8.13	Automated curve export from multiple result database files.....	8-52
8.14	Batch execution of solver processes .....	8-53
8.14.1	Batch solving trough the User Interface.....	8-53
8.14.2	Preparing for batch solving on remote computers .....	8-54
8.15	How to read error messages from the solvers .....	8-54
8.16	Using simulation events.....	8-56
8.16.1	Setting up simulation events.....	8-56
8.16.2	Event definition dialog.....	8-57
8.16.3	Simulation event properties.....	8-58
8.16.4	Solving multiple events and result organisation .....	8-58

## Chapter 9 Postprocessing Results

9.1	Postprocessing environment.....	9-2
9.2	Graphs .....	9-3
9.2.1	Creating graphs and curves .....	9-4

---

9.2.2	Showing a graph .....	9-6
9.2.3	Graph properties .....	9-6
9.2.4	Curve properties .....	9-7
9.2.5	Fourier analysis, differentiation and integration .....	9-14
9.2.6	Scale and Shift .....	9-16
9.2.7	Appearance.....	9-16
9.2.8	Curve Statistics.....	9-17
9.2.9	Fatigue calculation from standard S-N curves .....	9-18
9.2.10	View control .....	9-19
9.2.11	Export of Curve Data .....	9-19
9.2.12	Importing Curves and Graphs.....	9-21
9.2.13	Exporting to picture files.....	9-22
9.2.14	Printing graphs.....	9-22
9.3	Beam diagrams .....	9-22
9.3.1	Creating beam diagrams .....	9-22
9.3.2	Beam diagram curve properties.....	9-23
9.4	Graph groups.....	9-26
9.5	Animations .....	9-27
9.5.1	Managing animations .....	9-28
9.5.2	Animation properties.....	9-29
9.5.3	Available animation results .....	9-35
9.5.4	Performance of animation loading.....	9-37
9.6	Viewing animations .....	9-38
9.6.1	Play panel.....	9-39
9.6.2	Animation controls.....	9-40
9.6.3	Contour legend control.....	9-41
9.6.4	Exporting animations.....	9-44

---

## Chapter 10 Managing Results

10.1	Model and Result file handling .....	10-2
10.1.1	Discarding unsaved changes.....	10-2
10.1.2	Saving a model.....	10-2
10.2	Result File Browser.....	10-3
10.2.1	The Result File Browser dialog.....	10-3
10.2.2	Result manipulation .....	10-5
10.2.3	Result files from restart simulations.....	10-7
10.3	RDB directory structure .....	10-8
10.3.1	Part database.....	10-9
10.3.2	Response directory structure.....	10-9

## Appendix A FE Model Interface

A.1	Fedem Technology Link format .....	A-2
A.1.1	Syntax.....	A-2
A.1.2	Nodes.....	A-3
A.1.3	Structural elements.....	A-3
A.1.4	Properties .....	A-5
A.1.5	Loads .....	A-10
A.1.6	Strain Coat Elements.....	A-10
A.1.7	Strain Coat Properties.....	A-11
A.1.8	Other identifiers .....	A-12
A.2	Nastran Bulk Data File format.....	A-13
A.3	SESAM Input Interface File format .....	A-16

## Appendix B File Types and Usage

B.1	File types .....	B-2
B.1.1	Input files .....	B-2
B.1.2	Intermediate files .....	B-3

---

B.1.3	Results files .....	B-3
B.1.4	Other files.....	B-3
B.2	File usage for each program module.....	B-4

## Appendix C Command line options

C.1	Fedem GUI options .....	C-2
C.2	Reducer options (fedem_reducer) .....	C-4
C.3	Dynamics solver options (fedem_solver).....	C-7
C.4	Stress recovery options (fedem_stress) .....	C-15
C.5	Mode shape recovery options (fedem_modes).....	C-17
C.6	Strain rosette recovery options (fedem_gage). ....	C-19
C.7	Strain coat recovery options (fedem_fpp).....	C-21
C.8	Curve export options (fedem_graphexp).....	C-23

## Appendix D Beta feature documentation

D.1	Joints.....	D-2
D.1.1	Universal Joint .....	D-2
D.1.2	Constant Velocity Joint .....	D-2
D.1.3	Rigid Joint .....	D-3
D.1.4	Axial Joint.....	D-3
D.1.5	Free Joint .....	D-3
D.1.6	Prismatic Joint and Cylindric Joint .....	D-4
D.1.7	Cam Joint .....	D-5
D.2	Parts.....	D-6
D.2.1	Geometric stiffness.....	D-6
D.2.2	Component modes.....	D-6
D.2.3	Initial velocities.....	D-7
D.2.4	Stress recovery on element groups during dynamics simulation .....	D-7
D.3	Beams .....	D-8

---

D.3.1	Structural damping .....	D-8
D.3.2	Time-dependent stiffness scaling.....	D-8
D.3.3	Beam diagram curve domain adjustment.....	D-8
D.4	Beam properties .....	D-9
D.5	Springs.....	D-9
D.5.1	Sign-dependent stiffness scaling .....	D-9
D.5.2	Plastic springs .....	D-9
D.5.3	Spring failure .....	D-10
D.5.4	Stiffness proportional damping .....	D-10
D.5.5	Pulley element.....	D-10
D.6	Frictions .....	D-11
D.7	Additional masses .....	D-11
D.8	External loads.....	D-12
D.9	Sensors.....	D-12
D.10	Wave functions .....	D-13
D.10.1	Regular wave functions .....	D-13
D.10.2	Irregular wave functions .....	D-13
D.10.3	Embedded streamline functions.....	D-13
D.11	Result output control .....	D-14
D.11.1	Triads .....	D-15
D.11.2	Parts .....	D-15
D.11.3	Beams.....	D-15
D.11.4	Joints .....	D-15
D.11.5	Springs.....	D-16
D.11.6	Dampers .....	D-16
D.11.7	Loads (Force and Torque).....	D-16
D.11.8	Mechanism.....	D-17
D.12	Undocumented features .....	D-17

---

## Index

# Chapter 1 Introduction to Fedem

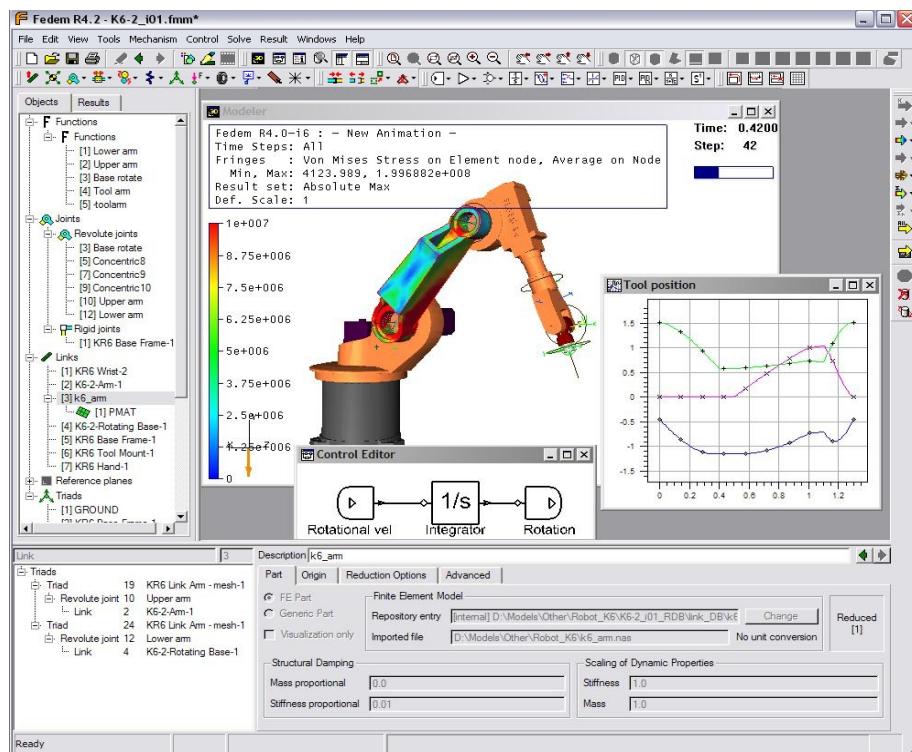
Welcome to Fedem! This chapter gives an overview of the Fedem software and its technological foundation. It explains what a Fedem model is, and outlines the different program modules.

Sections in this chapter address the following topics:

- [What is Fedem?](#)
- [Nonlinear structural dynamics](#)
- [What is a Fedem model?](#)
- [Control systems in mechanical analysis](#)
- [Using FE models](#)
- [Fedem solver modules](#)

## 1.1 What is Fedem?

Fedem, an acronym for **F**inite **E**lement **D**ynamics in **E**lastic **M**echanisms, provides both a technology platform and an engineering framework for virtual testing of complex mechanical assemblies. It provides a complete set of features to create, solve and post-process a model in a 3D graphical environment. Dynamics results in the form of curves and animations are available during and after model solution. Combined with the fast and numerically stable Fedem solvers, the user interface facilitates an engineering process with shortened turnaround times and quick access to simulation results for a clearer understanding of the physical behavior of the model. Fedem also provides intuitive and high-performing post-processing capabilities, including full stress analysis, eigenmode solutions, strain gage solutions and fatigue analysis for selected time steps.



## 1.2 Nonlinear structural dynamics

In multibody applications such as suspension systems, axle systems, car bodies, satellite appendages, industrial manipulators, medical equipment, high-speed mechatronic systems and so on, some of the mechanism components can be flexible and can experience large elastic deflections and coupling effects. To ensure sufficient accuracy, the simulation solver must account for the mutual dependencies between dynamic properties at the system level and structural flexibility at the component level. These requirements can be efficiently satisfied through a nonlinear structural dynamics approach.

In Fedem, a nonlinear structural dynamics approach is utilized in order to simultaneously solve structural deformations and 3D motion dynamics in the time domain. The mechanical assembly to be simulated is comprised of structural Parts, each represented by a linear elastic finite element (FE) model or a simplified stiffness description, and coupled together with linear or nonlinear joints. After a model reduction of each FE part based on a dynamic superelement formulation, the system equations are assembled and solved with respect to FE degrees of freedom (DOFs), allowing large translations and rotation due to a co-rotated theory.

## 1.3 Control systems in mechanical analysis

A mechanical system is often in a control loop including sensors measuring states in the mechanism, compensators representing control algorithms, and servo, hydraulic, or electrical actuators that generate the energy to drive the mechanism. These control modules and the mechanism must be integrated simultaneously in order to ensure sufficient accuracy of the simulation results.

To model your control systems, Fedem provides the Control Editor modeling environment, which combines the Fedem look-and-feel with graphical modeling tools similar to those of MATLAB®/Simulink®. These modeling tools are crucial when creating models of mechanical systems such as robots, milling machines, and space mechanisms. See [Section 2.4, "Touring the interface"](#) and [Chapter 5, "Control System Modeling"](#) for more information about the Control Editor.

## 1.4 What is a Fedem model?

A Fedem model is a virtual test model designed by you to simulate your mechanical systems. The virtual test model is an assembly of individual parts. The mechanical properties of the parts can be represented by either an FE model or a simplified stiffness description. You connect parts, virtual joints, springs, and other elements to create an accurate virtual test model of a movable mechanical system or mechanism.

In order to minimize the time needed to calculate and simulate mechanisms, a model reduction process is used to reduce the FE models of the mechanism into superelements with external nodes. These external nodes are defined at FE nodes that serve as connections between the various model entities in the assembly.

Once constructed, a Fedem model retains the FE characteristics of its component parts, and can therefore be treated as a geometrically nonlinear FE model. The mechanism is then allowed to experience large translations, rotations, and nonlinear, behavior-dependent loads in the simulation of its dynamic motion.

Functions, measured time history input data and control systems can be used to model virtual test events by controlling loads, motion, spring lengths, etc.

The Fedem solvers can then calculate the motion kinematics, dynamics, structural flexibility, stresses, strains, and varying loads in one consistent model.

## 1.5 Using FE models

Fedem imports FE models created in external systems by interfacing with the MSC.Nastran® Bulk Data File (.bdf and .nas) format (see [Section 5.1.2, "Creating parts by file import"](#)). Many FE modeling systems (including I-DEAS®, Pro/ENGINEER®, Altair® HyperMesh®, NEiWorks/NEiFusion and MSC/Patran®) can export data to the Bulk Data File format, enabling you to use your favorite FE modeling program with Fedem.

Fedem can also import models which are saved using the SESAM input interface file (.FEM) format.

Each FE model must include descriptions of its nodal coordinates, element topologies, element properties, material data, nodal attributes, and so on.

The results obtained in the simulation depend to a large extent on the FE models used. To create a satisfactory model of a real structure, the analyst must combine insight into the nature of the problem, experience with the Finite Element Method (FEM), and knowledge of the general rules of FE modeling.

## 1.6 Fedem solver modules

Fedem can use a set of separate program modules to perform the different type of calculations on the model. Thus, there are modules for FE model reduction, dynamic mechanism simulation, stress and mode shape recovery, strain rosette analysis and strain coat analysis. The main Fedem application is a graphical user interface that manages the execution of each solver module. However, they may also be run separately as batch processes, or remotely through cloud services. The Fedem solver modules are described briefly below.



**Tip:** To display a list of all the modules and their version and build date, press the **About Fedem...** entry in the Help menu.

### 1.6.1 FE Part Solver

The FE model Solver performs a linear analysis of an individual FE part, subjected to loads and boundary conditions defined in the FE model file discussed above in [Section 1.5, "Using FE models"](#).

This module is mainly intended as a model debugging tool, where you can study the behavior of an FE part without the need of having it integrated in a complete mechanism model.

You can also perform *Stress Recovery* on an FE part, based on the displacement state computed by the FE model Solver.

### 1.6.2 Reducer

The FE model Reducer performs a superelement reduction of the FE model representing the mass and stiffness of a part. A superelement reduction reduces the required DOFs to a minimum. The retained DOFs, also called external DOFs, are the results of a dynamic superelement reduction technique called Component Mode Synthesis reduction, also known as CMS reduction.

More information on this topics can be found in FEM textbooks.

### 1.6.3 Dynamics Solver

The Fedem Dynamics Solver performs a nonlinear dynamics simulation. This means a simulation of the motion and deformations of the superelements and the joints, as they respond to load and displacement time histories and Control System output.

At any time step of the simulation, the model can be linearized whereby an eigenvalue analysis can be performed.

Part deformations, von Mises stresses, and strain gage results may optionally also be computed during the dynamics simulation.

### 1.6.4 Stress Recovery

The Stress Recovery module recovers the internal DOFs from the deformations of the external DOFs simulated by the *Dynamics Solver*. The element stresses, strains and beam forces are then calculated.

### 1.6.5 Mode Shape Recovery

The Mode Shape Recovery module recovers the mode shapes of each FE part from the Eigenvalue results of the *Dynamics Solver*.

### 1.6.6 Strain Rosette Analysis

The Strain Rosette Analysis module recovers the stresses and strains on virtual strain gages on the FE parts, based on the *Dynamics Solver* results. The output is time history data of stresses and strains similar to output from real strain gages.

### 1.6.7 Strain Coat Analysis

The Strain Coat Analysis module recovers the stresses and strains on the strain coat elements in the model, and calculates a summary of the recovered results over its entire time history. The output from this analysis is maximums of certain stress/strain quantities over time, and other quantities to assess damage and life time of the FE part.

### 1.6.8 Curve Export Utility

The Curve Export Utility module allows you to automatically export a set of curves to a single ASCII/RPC-file. The result data on which the curves are defined can be distributed on several results database files. This program module can only be run separately as a batch process.

# Chapter 2 Learning the Basics

This chapter outlines system requirements, explains where models and results are saved, and introduces the Fedem user interface and common commands.

Sections in this chapter address the following topics:

- [System requirements](#)
- [Storing models and results](#)
- [Starting Fedem](#)
- [Touring the interface](#)
- [Executing commands](#)
- [Visualizing the model](#)
- [Opening and saving model files](#)
- [Loading and unloading FE-Data](#)
- [Exporting objects](#)
- [License information](#)
- [Customizing Fedem using plug-ins](#)
- [Customizing Fedem using Addons](#)

## 2.1 System requirements

The following are minimum system/hardware requirements and recommended settings for optimal Fedem performance.



**NOTE:** Use of resolution settings other than those recommended below may require you to move or resize windows and dialog boxes to fit your screen.

**Table 2-1: Windows 2000/XP/Vista/W7-W11 and recommended settings**

<b>Processor</b>	Recommended: Pentium 4 @ 2 GHz. Minimum: Pentium III @ 850 MHz.
<b>RAM</b>	Recommended: 2 GB. Minimum: 256 MB.
<b>Free disk space</b>	Dependent upon the size and complexity of mechanism models.*
<b>Graphics card</b>	Recommended: high end graphics card. Required: OpenGL® API support.
<b>Pointing device</b>	Three-button mouse with wheel.
<b>Resolution setting</b>	1,280 x 1,024 pixels or higher.

\* Larger models may require several GBs.

## 2.2 Storing models and results

Fedem uses the following files and directories to store the contents of a Fedem model:

- The mechanism assembly description and all simulation parameters are saved in the Fedem Mechanism Model (*.fmm*) format. This file is called the model file and contains the complete description of your model except the FE data files and the simulation results.
- FE data files and all simulation results are saved in a directory named *<modelfilename>\_RDB* (the model name specified by the user is substituted for *<modelfilename>*). Fedem creates this directory in the same location as your model file. Within this directory, the FE data files are stored in a directory named *link\_DB*, unless a *FE model repository* is used (see [Section 5.1.6, "Using FE model repositories"](#)).

[Appendix B, "File Types and Usage"](#) contains more information about Fedem file types, whereas the directory structure is shown in [Section 10.3, "RDB directory structure"](#).

### 2.2.1 FTL format

The Fedem Technology Link (`.ftl`) format is used to store FE models as sets of data in text format (see [Section A.1, "Fedem Technology Link format"](#) for more information about the `.ftl` format).

### 2.2.2 FTC format

The Fedem Technology Cad (`.ftc`) format may be used to store CAD geometry in text format.

### 2.2.3 Other supported formats

Fedem can import FE models (see [Section 5.1.2, "Creating parts by file import"](#)) in the MSC.Nastran Bulk Data File (`.bdf` or `.nas`) format (see [Section A.2, "Nastran Bulk Data File format"](#)) and the SESAM input interface file (`.FEM`) format. Earlier versions of Fedem used the Fedem Link Model (`.f1m`) format for FE models. These files can still be imported and saved to the `.ftl` format.

Fedem can also import CAD geometry from VRML files (`.wrl`, `.vrm1`, `.vr1`, `.wrz`) for visualization of *Generic Parts* (see [Section 5.1, "Parts"](#)).

## 2.3 Starting Fedem

Fedem can be started from either the Windows *Start* menu, the Windows file browser, or the command prompt.

- To start Fedem with a new, empty model, press the Fedem icon on the Windows Desktop or choose **Fedem** from the Windows *Start* menu.
- To start Fedem and open an existing model, double-click the icon representing the desired model file in the Windows file explorer.
- Open a command prompt, navigate to the installation directory of Fedem and type `fedem` optionally followed by some command line options (see [Section 2.3.1, "Command-line options"](#)).
- To open an existing model from the command prompt, navigate to the installation directory and type `fedem -f <modelpath>.fmm` substituting `<modelpath>` with the path and base name of your model file. Alternatively, navigate to the directory of the model file and type `<installationdir>/fedem -f <modelfile>.fmm` where `<installationdir>` is replaced by the directory name of the Fedem installation, and `<modelfile>` is the base name of the model.

When Fedem is started with a new, empty model, the model file is named *untitled\_<#>.fmm*, where *<#>* indicates a unique running number. This empty model file will contain only some default settings, animations and graphs. The file will be created in your default login directory when Fedem is started from the Desktop icon or from the Start menu.

### 2.3.1 Command-line options

Fedem has several command-line options that can be used to achieve different tasks or settings. These options can be invoked when starting Fedem from the command prompt. Among others, you have options for starting Fedem in batch mode without any graphical user-interface at all, which is useful when you just want to run an existing model without modifying it. Refer to [Section C.1, "Fedem GUI options"](#) for the complete list of available command-line options.



**TIP:** You can create a new model file with a specific name by specifying a path and filename that does not yet exist, using the *-f* option. The file will be created for you and opened when Fedem starts.

### 2.3.2 Template model file

The default settings of a new model are stored in a template model file which is loaded whenever a new model is created (either when Fedem is started, or when using the **New** command, see [Section 2.7.3, "Starting a new model!"](#)). The template file can be modified or changed as needed, and is located in *<installationdir>/Templates/default.fmm*.

If a different location for the template file is wanted, the environment variable **FEDEM\_TEMPLATE\_FILE** can be defined to contain the desired path to the template file.

#### Predefined template files

The *Templates* directory of the Fedem installation contains three files; *default.fmm*, *default\_m.fmm* and *default\_mm.fmm*. The files *default.fmm* and *default\_m.fmm* are identical and suitable for models using SI unit set, while *default\_mm.fmm* contains default settings that are suitable for models using millimeter, newton and megagram as unit set. Simply copy the file *default\_mm.fmm* to *default.fmm* if you prefer mm/N/Mg as your default unit set.

### 2.3.3 Console window

When Fedem is started on a Windows system, a separate console window can be opened in addition to the Fedem main window, by specifying the command-line option `-console`. This window lists some startup messages, and may also contain some low-level error messages during a Fedem session, that normally can be ignored by users, but sometimes might help understanding an abnormal incident.



**WARNING!** Once opened, the separate console window must remain open throughout the whole Fedem session; closing it explicitly closes the entire Fedem session immediately, without the option to save unsaved work.



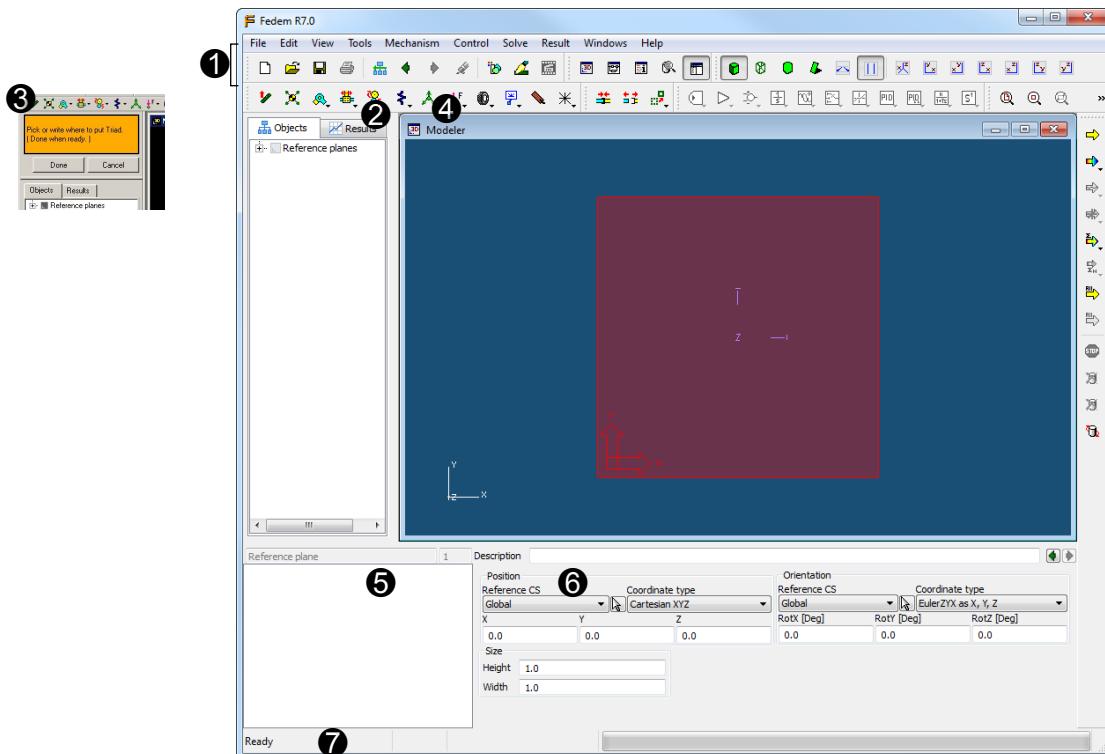
**TIP:** If abnormal behavior occurs, or if Fedem stops responding, check the console window (if activated) and the output list (discussed later) for important information. Take note of any errors or warnings before contacting Fedem technical support ([support@fedem.com](mailto:support@fedem.com)).

After finishing some initial setup tasks, the main window of the Fedem graphical user interface is displayed. In the following sections, this graphical user interface is described.

## 2.4 Touring the interface

### 2.4.1 Main window

Starting with an empty model file, the Fedem main window appears as shown below.



- ① **Menus and tool bars** – contain buttons used to initiate commands.
- ② **Model Manager panel** – contains the *Objects* and *Results* tabs, which allow you to create, manage, and delete the objects in your model and define animations and graphs of your results.
- ③ **Guide panel** – This orange panel may pop up above the Model Manager panel when you invoke a command. It will tell you what you are expected to do during the different steps of the command. It also provides **Done** and **Cancel** buttons used to accept choices or to cancel the command, respectively.
- ④ **Workspace area** – contains the *Modeler*, *Control Editor*, and *Graph* views for constructing and viewing models and results.

- ⑤ *ID and Topology panel* – contains a list of objects related to the selected item.
- ⑥ *Property Editor panel* – allows you to view and edit the properties of individual objects in your model.
- ⑦ *Status bar* – provides information of the status, progress information and whether some solver is running.

## 2.4.2 Menus and tool bars

Fedem commands are initiated from buttons on the tool bars and menus. All Fedem commands can be accessed from the menus, while the tool bars display only some of the most commonly used commands. The menus are arranged from left to right in logical order of task execution, starting with standard file functions, then continuing with viewing commands, mechanism modeling, control system modeling, analysis/solution tools, and finally results management.

2

### Command sensitivity

Menu and tool bar buttons are sensitive to the active view and object selection. For example, if an object in the *Modeler* view is selected, the *Graph* view controls appear dimmed (grayed-out) and cannot be selected.

### Fedem tool bars

Fedem uses the following tool bars (a tool bar handle to the left (or on top) marks the beginning of a tool bar):

- *Standard*: provides standard file operations such as **Open**, **Save**, **Exit**, and selecting and deleting objects.
- *Windows*: provides commands for controlling the active view selection (*Modeler*, *Control Editor*, *Output List*, *Result File Browser*) and hiding/showing the Model Manager and Property Editor panels.
- *Zoom and Pan*: provides commands for zooming and panning the two-dimensional display of graph views. (See [Section 2.6.4, "Zoom and Pan"](#) for more information about these commands.)
- *3D View Control*: includes commands for rotating the view and changing the view perspective in the *Modeler* view. (See [Section 2.6.3, "3D View controls"](#) for information about manipulating the view.)
- *Solvers*: these commands are used to set up, start, and stop mechanism analyses and calculate specific results. (See [Chapter 8, "Mechanism Analysis"](#) for information about these commands.)

- **Mechanism Creation:** contains commands for importing parts and creating mechanism entities such as joints, springs, dampers, forces, and sensors (see [Chapter 5, "Mechanism Elements"](#)).
- **Mechanism Tools:** provides commands used in modeling such as moving, attaching, copying objects, and applying motion constraints. (See [Chapter 5, "Mechanism Elements"](#) for information about modeling tools.)
- **Control Creation:** provides commands for creating control objects. (See [Chapter 7, "Control System Modeling"](#) for detailed descriptions of control objects.)
- **Control Tools:** contains commands for manipulating the control system. (See [Chapter 7, "Control System Modeling"](#) for information about modeling control systems.)

### Manipulating tool bars

Only some of the commands accessible from the tool bars are displayed. You will see that there are arrows (▼) beside some of the icons. To access other commands, click and hold down the ▼ button. A menu of additional, related commands is displayed (example at right). Selecting a command from the menu initiates the command and replaces the button's function with that of the new command. You can select another option at any time by clicking, holding down the ▼ button, and selecting a different command.



Fedem provides several ways to manage tool bars:

- Right-click a tool bar handle and select an option to relocate, show, or hide the tool bar.
- Right-click an empty space in the tool bar area and select an option from the list to show or hide a tool bar.
- Double-click a tool bar handle to show or hide the tool bar.
- Drag a tool bar handle to the left, right, top, or bottom of the main window to relocate the tool bar.

### 2.4.3 Model Manager

The Model Manager panel contains lists of all the objects and result views that make up your model. This includes mechanism, modeling, and control system objects, along with animations and graphs. In each list, objects are grouped by type and sorted by identification numbers or names. In the Model Manager panel, right-click menus can access several commands that can be applied to the selected objects.



**NOTE:** The Objects and Results lists are empty (or nearly empty) until you create items.

#### Selecting items

In the Model Manager panel, you can select items in several ways:

- Highlight a single item.
- Hold down the **Shift** key and click multiple items.
- Hold down the **Ctrl** key and click multiple items (or a single item to select/deselect).
- Click and drag the mouse over multiple items.

#### Deselecting items

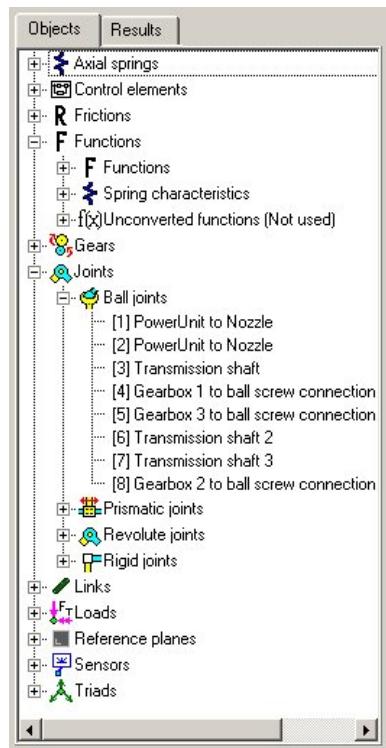
To deselect all items, right-click an empty space in the Model Manager panel.

#### Sorting

The objects in the Model Manager panel can be sorted, either by ID-number or by item name. To switch sorting mode, right-click inside the panel and select either **Sort by ID** or **Sort by Name** from the menu that appears. By default, the sorting is based on ID-numbers.



**TIP:** Right-clicking in the Model Manager panel will display a pop-up menu with commands that applies to the current selection. The contents of this menu depend on the type of the selected object.



## Objects

The *Objects* tab displays a list of all modeling objects in your model. Selecting an item from this list highlights it in the *Modeler-* or *Control Editor* view and displays its properties in the Property Editor panel.

## Results

The *Results* tab displays a list of the result views you have created. The available result views are graphs which contains curves (individual sets of plotted data), and animations.

Selecting a graph from this list will cause the view containing the selected graph to pop up, if loaded. Selecting a curve will highlight it (the curve is rendered red) and raise the graph view in which it resides as well.

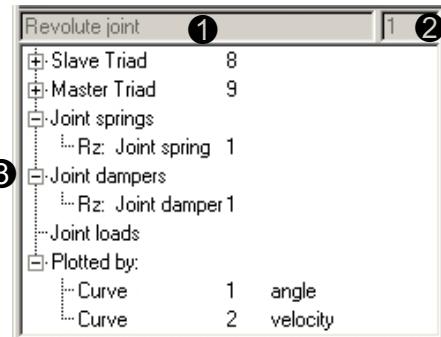
The properties of the different result objects are shown in the Property Editor panel when they are selected.

See [Chapter 9, "Postprocessing Results"](#) for more information about graphed and animated results.

### 2.4.4 ID and Topology panel

When an object in the model is selected, the ID and Topology panel displays information related to the object as described below. If multiple items are selected, only the item selected last is displayed.

- ① *Item Type* – the type of the selected object (for example, revolute joint, gear, or spring).
- ② *ID Number* – a unique integer that distinguishes one item of the same type from another.
- ③ *Topology view*– a list that displays the mechanism objects related or connected to the selected item.



**TIP:** The *Plotted by* branch in the *Topology view* lists all curves that plot result quantities from the selected structural object.

ID numbers are assigned automatically to new objects in numerical order. If you delete an object such as a ball joint with ID Number 3, this number is free and the next ball joint created is then assigned ID Number 3.

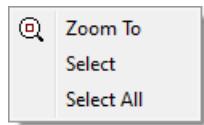
## Topology view and browsing

Most mechanism components are related to other objects; for example, joints consists of triads that are connected to parts, and sensors are measuring variables from other mechanism components.

These relations are shown by the *Topology* view in a hierarchical fashion, indicating their topological relationships. This list can then be used to investigate these related objects, and to browse through the mechanism model through the topological connections. This can be done by using the browsing features offered by the *Topology* view described below.

**Temporary highlight:** The items in the *Topology* view will be highlighted in the *Modeler* view when you select an item keeping the mouse button pressed. This is useful to see exactly which object in the 3D view that corresponds to the listed item.

Right clicking an item in the *Topology* view will show a tiny pop-up menu that allows you to either **Zoom To** or **Select** the item, or to do a **Select All**.



**Zoom To:** This command zooms to the item, making it easy to locate it in a complex model. See also [Section 2.6.4, "Zoom and Pan"](#).

**Select:** This command will select the right clicked object from the *Topology* view, and thus jump to it showing the properties of that item instead. See also [Section 2.5.1, "Select"](#) on how to get back to your previous selection.

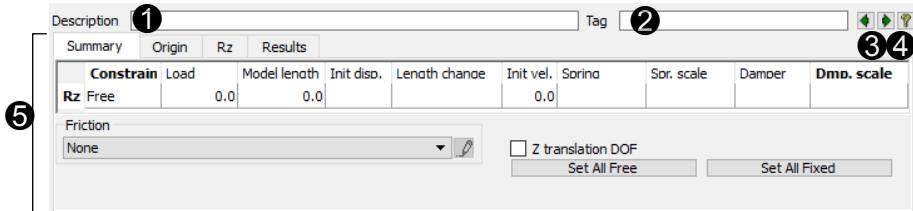


**TIP:** Double clicking an item in the *Topology* view will also select it.

**Select All:** This command will select all top-level objects currently in the *Topology* view, but showing the properties of the last object only. The effect of this command is the same as doing a multi-selection from the *Objects* list view of Model Manager panel using the **Shift** key, see "[Selecting items](#)" in [Section 2.4.3](#).

### 2.4.5 Property Editor

The Property Editor panel is used to view and edit the properties of mechanism items. The appearance of properties is different depending on the object selected. The image below is an example showing properties that are common to some of the Fedem modeling objects.



- ① *Description* – an optional user-supplied name or identifying remarks
- ② *Tag* – an optional identifier which can be used to refer to the object from python scripts using the *fedempy* interface
- ③ – buttons for navigating back and forth through the latest selections
- ④ – help button that opens the Reference guide on the Property Editor panel content, depending on the type of the selected object
- ⑤ *Properties* – editable attributes specific to the selected object



**NOTE:** The description field may contain any ASCII-character, except for the “-character which is reserved for text string delimiters in the model file.



**NOTE:** The description field is also used to activate beta features, see [Appendix D, "Beta feature documentation"](#).



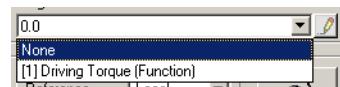
**CAUTION:** After editing a value in the Property Editor panel by typing, you must press the **Enter** key to apply the change.

### Property menus

Many mechanism items have internal properties that can depend on the simulation time, measurement of a system variable or some internal variable in the item in question. Property menus are used to set up such properties in a simple way. These menus consist of an option menu that in some cases are editable, and an **Edit** button.



The menu contains references to other entities in your model that can be used as input for the property in question. A non-linear spring characteristic (e.g., a force-displacement curve) will be listed in the stiffness property menu of an axial spring, functions and control outputs will be listed in the magnitude property menu of a load etc. Press the menu button to access the list.





By pressing the **Edit** button you will select the item shown in the menu, as if you had selected it in the Model Manager panel. This is a convenient way of navigating through the relations of the model, giving a simple way of finding the details of a complex model.

In some of the property menus, a numeric value can be entered instead of selecting a reference from the menu. The spring stiffness property is a good example. Entering a numeric value will assign that value as a constant spring stiffness to the spring in question.



**TIP:** *Property menus that accept a numeric value always have a numeric value as default, while "None" is normally the default for Property menus not accepting a number.*

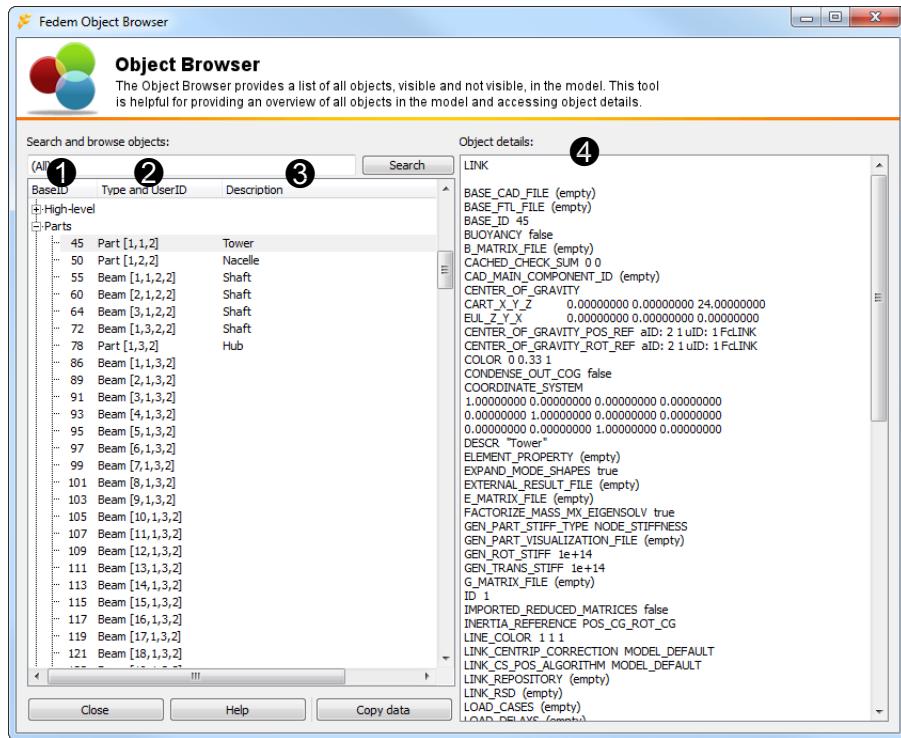
To change a property from referring to a constant, select the top entry in the pull-down list (which is either *None* or the last numerical value that was entered in the same box), or delete the current contents of the box, type in a numerical value and then press **Enter** to apply the change.

#### 2.4.6 Object Browser



An alternative tool to display (but not edit) the detailed properties of mechanism objects is via the Object Browser dialog box. It is opened by selecting **Objects Browser...** from the *Tools* menu or from the right-click menu in the *Objects* list view of the Model Manager panel.

The Object Browser dialog box is shown below.



- ① **BaseID** – This column lists the base ID of the objects. These are unique ID numbers over all objects in the model that are used for internal book-keeping only. They are not visible in the Fedem GUI elsewhere.
- ② **Type and UserID** – This column lists the type and user ID of the objects, which are used as unique identification of the objects elsewhere in the Fedem GUI. The user ID consists of a comma-separated list of values in brackets [] when the object is part of a higher-level sub-assembly, see [Section 5.15.4, "User ID convention in assemblies"](#).
- ③ **Description** – This column lists the description string of the objects.
- ④ **Object details:** – This field lists in plain text all the information stored for the selected object in the model database. The formatting of this list resembles that of the model file (.fmm) itself. The first line contains the model file keyword for the selected object. Then the field name (in upper case) and value of each data field of that object (including also the base- and user ID and description found in the first three columns) are listed in alphabetic order of the field name.



**TIP:** The list in the left half of the Object Browser can be sorted with respect to the base ID, Type and User ID, or Description, by clicking on the respective column headings.

When opened, the Object Browser dialog box will display details for the currently selected object in the model. You can also search for objects by typing their description in the search field and clicking the search button. The Object Browser will list all objects in the model if the search field is blank, or if no object was selected when the dialog box was opened.



**NOTE:** The Object Browser dialog box displays only the objects that affect the numerical simulation of the model. Therefore, any result objects (graphs, curves and animations) as well as the higher-level sub-assembly objects are not listed.

The main purpose of the Object Browser dialog box is to provide an overview of the model-file keywords and field names used for the various object types, as these are necessary for setting up the simulation events when, for instance, a model consists of multiple load cases. See [Section 8.16, "Using simulation events"](#).

#### 2.4.7 Workspace

The Workspace area is used for constructing, manipulating, and viewing mechanism models, control systems, graphed results, and animations. The Workspace can contain several views, including the *Modeler*, *Control Editor*, and multiple views for graphed results. These views are described in the following sections.



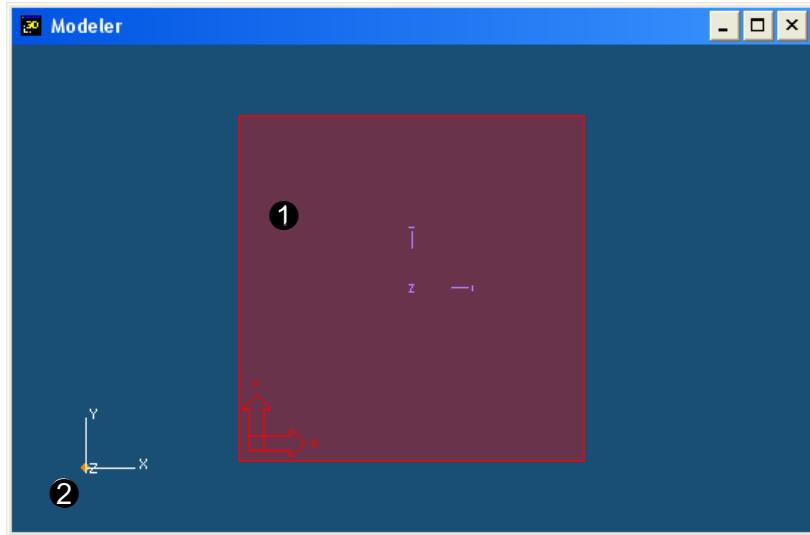
**TIP:** The views in the Workspace can be managed using the **Tile** and **Cascade** commands from the Windows menu.

##### Modeler

The *Modeler* view displays a 3D rendering of your mechanism and provides dynamic viewing capabilities such as zooming, panning, and 2- and 3-dimensional rotation (see [Section 2.6.1, "3D Navigation"](#)). This view is also used to show animated simulation results. Select the *Modeler* view to view, create, or edit a mechanism model.

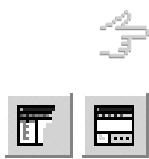


To open the *Modeler* view, select **Show Modeler** from the Windows tool bar or menu. The *Modeler* view is displayed as shown below. See also [Section 4.2.1, "Modeler view"](#).



- ① **Reference Plane** – The shaded area in the center of the *Modeler* view represents a plane, which can be considered the ground or base for your models.
- ② **Global Directions** – The arrows located in the lower left corner of the *Modeler* view show the orientation of the global coordinate system and the direction of the gravity vector,  $\mathbf{g}$ .

**TIP:** The Modeler view can be expanded to almost full screen size by hiding the Model Manager and Property Editor panels. To hide these panels, click the **Model Manager** and **Property Editor** buttons on the Windows tool bar (or the View menu). Hiding the tool bars also increases the viewing area of the Modeler view (see "[Manipulating tool bars](#)" in [Section 2.4.1](#)).

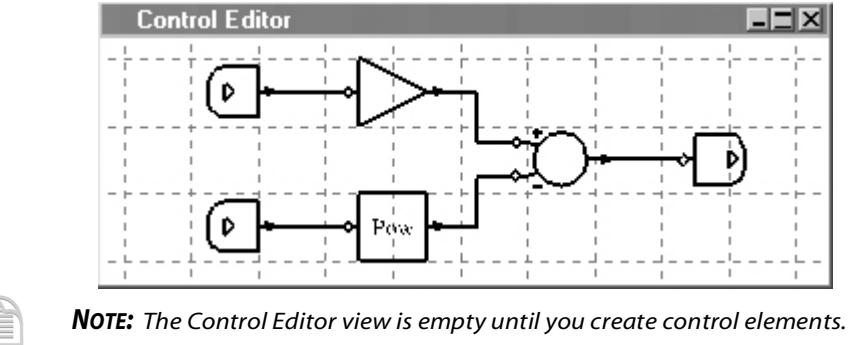


### Control Editor

The *Control Editor* view is a workspace for editing the control system. The graphical representation is a block-based diagram that consists of a series of control blocks that can be connected to simulate your control system. This editing environment allows you to create and manipulate the control system using drag-and-drop functionality. It also features grid and snapping tools (see [Section 7.4.1, "Setting Grid and Snap"](#)).



To open the *Control Editor* view, click the **Show Control Editor** button on the Windows tool bar (or from the Windows menu). The *Control Editor* view displays the control system (an example is shown below). See [Chapter 7, "Control System Modeling"](#) for more information on the topic.



**NOTE:** The Control Editor view is empty until you create control elements.

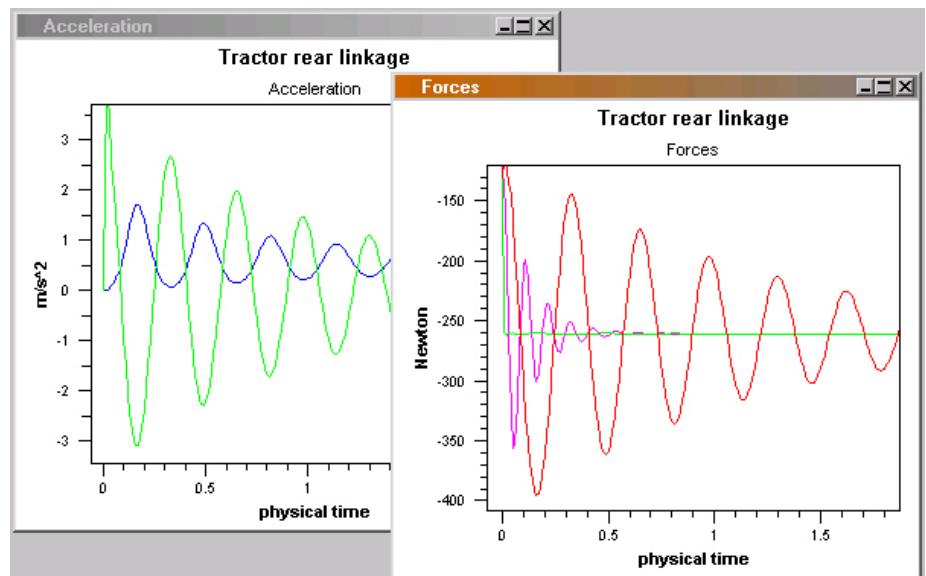
2

### Graph Views

The *Graph* view can display various graphed results. You can customize graphs of selected simulation results and manipulate the *Graph* view.



To open a graph view, right-click the graph from the *Results* list of the Model Manager panel, and select **Show Graph**. The graph views are displayed in the Workspace area as shown below.

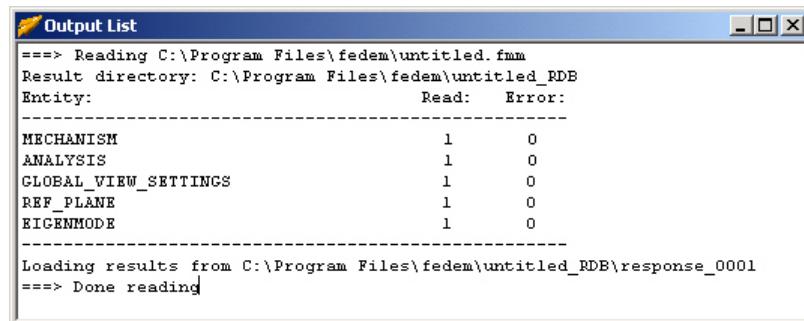


## 2.4.8 Output List

The *Output List* view displays written output from Fedem, such as a log of commands and solution, and error messages. This view allows you to observe the commands performed by Fedem.



To open the *Output List* view, select **Show Output List** from the *Windows* menu or tool bar.



```

Output List
====> Reading C:\Program Files\fedem\untitled.fmm
Result directory: C:\Program Files\fedem\untitled_RDB
Entity:           Read:   Error:
-----
MECHANISM          1      0
ANALYSIS           1      0
GLOBAL_VIEW_SETTINGS 1      0
REF_PLANE          1      0
EIGENMODE          1      0
-----
Loading results from C:\Program Files\fedem\untitled_RDB\response_0001
====> Done reading

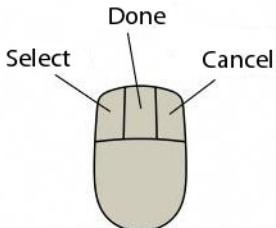
```

The text in the *Output List* view is also written to a log file. The name of this log file is the same as the current model file name, but with extension *.log* instead of *.fmm*. Therefore, a new log file is opened whenever you **Open** a new model, or perform a **Save As...** command.



**NOTE:** If a log file already exists for the model you open from an earlier Fedem session, the output from the current session is appended to that file. That means that the entire history of the Output List content for the model is recorded. In addition, the date and time of the model open and close operations, as well as the Fedem version used, are recorded to the log-file.

## 2.5 Executing commands



When performing commands in Fedem, the Guide panel prompts you with instructions for completing each command. You may be asked to select mechanism objects or locate points to place or move objects. Performing commands makes use of three actions: **Select**, **Done**, and **Cancel**.



### 2.5.1 Select

To select items in your model, or points on objects as references for moving or creating items, place the cursor over the object or position and press the left mouse button (left-click). The item is highlighted and its properties can then be edited in the Property Editor panel.



**NOTE:** Some commands require that an object is selected from views in the Workspace area only, such as the Modeler or Control Editor views. Instructions regarding these commands are provided in the Guide panel.



**TIP:** To deselect an item, simply click an empty space within the Modeler view.

#### Snapping

When selecting objects in your model, the selection automatically snaps to a point on the object such as the nearest node on an FE part, the center point of a joint, and so on. This makes your selection quick and accurate.

Snapping behaves differently on different types of objects. FE parts, VRML models, CAD parts and mechanism symbols all have different snapping policies.

Selection snaps to FE nodes on FE parts, to vertices on a VRML part, to geometry features such as center points and edges on CAD parts, and to important points on mechanism symbols.

#### Multiple selection

Some Fedem commands, such as **Smart Move** and **Delete**, allow you to select several items at once. To select more than one item, press and hold down the **Ctrl** key and then click the items you want to add to your selection.

If you accidentally add the wrong object to the selection, simply release the **Ctrl** key and click an empty space within the *Modeler* view. The last selected item is deselected.

#### Selection history

Fedem maintains a history of the items you select in the current session. This history can be accessed using the **Select Backward** and **Select Forward** commands.



- To choose a previous selection, press the **Select Backward** button. You may need to press it several times to cycle back through the selections until the desired object or selection is reached.



- To select a recent selection, press the **Select Forward** button once or as many times as necessary until the desired selection is reached.

### Co-located items

Sometimes several items in a model are located very close together or on top of each other; in Fedem, these are called *co-located items*. To select a co-located item, click the same spot several times to cycle through the items. Fedem cycles from the item closest to the viewer to the one furthest from the viewer.

### Selection Filter

Some Fedem commands allow you to select only certain types of items. These restrictions are automatically imposed and based on the type of command in use. For example, sensors cannot be applied directly to parts, and Fedem will therefore limit your selection to other types of mechanism items. To make the selection even easier, you can also filter the selectable items by limiting the types of items displayed in the *Modeler* view.



**TIP:** To limit the display of mechanism objects, click the **General Appearance** button on the Standard tool bar, then disable Mechanism Symbols as necessary. (See also [Section 2.6.5, "General Appearance"](#).)

### Selecting ground

To select the ground during modeling, simply click anywhere on the Reference Plane.

## 2.5.2 Done

When executing a command in Fedem, the Guide panel prompts you to select items or locate points. When you have achieved the desired selection, press the **Done** button in the Guide panel to accept it.



**TIP:** You can also press the center mouse button within one of the Workspace views or the **Enter** key on the keyboard to accept the selection.

## 2.5.3 Cancel

To abort or escape a command procedure, press the **Cancel** button on the Guide panel.



**TIP:** You can also press the right mouse button within one of the Workspace views or the **Esc** key on the keyboard to cancel a command.

## 2.6 Visualizing the model

### 2.6.1 3D Navigation

The 3D navigation commands enables you to change the view without interrupting the current command or procedure. There are several different sets of viewpoint control commands:

- Middle mouse button/wheel + mouse motion
- Function keys (**F1**, **F2**, **F3** and **F4** + mouse motion)
- Keyboard keys
- Predefined view tool buttons

2

#### Middle mouse button/wheel

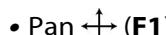
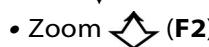
Pressing and holding the middle mouse button (or wheel if your mouse have that) while moving the mouse will rotate the view around the rotation center. If the button/wheel was pressed near the edge of the 3D view, the rotation will be restricted to the viewport's normal axis.

Rolling the mouse wheel will zoom in and out. When the mouse wheel is used to zoom in, the view will be zoomed towards the position of the mouse pointer giving a combined zoom and pan behavior.

Using the middle mouse button commands is the most used 3D navigation method in Fedem.

#### Function keys

The function key based navigation can be used when you need added control over the navigation. The commands include:

- Pan  (**F1**)
- Zoom  (**F2**)
- Rotate  (**F3**)
- Select Rotation Center  (**F4**)

These functions are often useful while working in the *Modeler* view. We recommend that you keep your left hand near these function keys while you work.

To use the function key commands, press and hold the function key, and move the mouse to manipulate the view. The manipulation will only

occur as long as the mouse is inside the *Modeler* view. By pressing the left mouse button, you may avoid this restriction.

When the function key is released, the manipulation stops.



**CAUTION:** When pressing the left mouse button while using the function keys, Fedem grabs the mouse and keyboard control.

### Pan - (F1)

The Pan command shifts the view left, right, up or down.

### Zoom - (F2)

The Zoom command moves the scene closer or further away from the camera. It pays attention to the rotation center, and will zoom towards it (see "[Select Rotation Center \(F4+Select\)](#)" below). This is useful when you need to examine an object or its components closely.



**TIP:** To achieve maximum zoom at a specific point, select the point using **Select Rotation Center (F4)** and then zoom in on the point using **Zoom (F2)**.

### Rotate (F3)

The Rotate command enables you to dynamically rotate your model around a point or an axis at the rotation center (see "[Select Rotation Center \(F4+Select\)](#)" below). The rotation can be performed in two different ways, depending on the position of the cursor when you press **F3**.

- Axis rotation: With the cursor near the edge of the *Modeler* view, the model rotates around an axis that is perpendicular to the screen.
- Point Rotation: With the cursor near the center of the view, the model rotates around a point located at the rotation center some distance into the model. This allows rotation of the model in any direction around the point.



**TIP:** The rotation motion is sensitive to the speed of the mouse. If the mouse is moved slowly the control of the rotation becomes finer and makes it possible to accurately control the view along long constructions.

### Select Rotation Center (F4+Select)

The **Select Rotation Center** command enables you to select a new center for zooming and rotation. When you use **F4** to select a point, the selected target point shifts to the center of the view and becomes the new dynamic center used by the **Zoom (F2)** and **Rotate (F3)** commands.

This target point remains the dynamic center until the model is moved by some other viewing command.

To select a new rotation point, press and hold down the **F4** key, move the cursor to the target point, and click the left mouse button.



**TIP:** To closely examine a part of your model, set the dynamic center (**F4**) at the point of interest, and use **Zoom (F2)** to magnify the view. You can then easily examine the point from many directions using **Rotate(F3)**.

### Keyboard keys

To rotate the model by increments of 15 degrees, use the arrow keys. Rotation and panning may also be done by pressing **Shift** (rotate 90 degrees), **Alt** (rotate around screen normal) or **Ctrl** (panning) in combination with arrow keys. To zoom in and out, press **z** or **Shift + z**.

## 2.6.2 Predefined view tool buttons



### Isometric

To display an isometric view of your mechanism, click the **Isometric** button.



### Top

To display the top view of your mechanism, click the **Top** button.



### Right

To display the side view of your mechanism, click the **Right** button.



### Front

To display the front view of your mechanism, click the **Front** button.



### Bottom

To display the bottom view of your mechanism, click the **Bottom** button.



### Left

To display the left side view of your mechanism, click the **Left** button.



### Back

To display the back view of your mechanism, click the **Back** view button.

### 2.6.3 3D View controls

Fedem provides several 3D viewing commands for use in the *Modeler* view. The following commands can be accessed on the *3D View Control* tool bar (or *View* menu):



#### Solid View

To display all mechanism items as solid/shaded objects, click the **Solid View** button. This is on by default.



#### Line View

To speed up graphic performance and display all mechanism items as outlines, click the **Line View** button.



#### Flat Colors

To render the model without shading, click the **Flat Colors** button. This is especially useful when viewing color plots. This option is off by default.



#### Show Top Faces

To distinguish the top and bottom faces of shell elements, click the **Show Top Faces** button. The top faces will then be rendered normally while the bottom faces will be rendered dark/black. If some of the parts have the detail level set to Reduced Surface or Reduced Surface and Internals, only a rough indication on the states of the faces is given. To see the exact top and bottom of every face on a part, set the detail level to Surface or Surface and Internals. This is off by default.



#### Perspective

To display a perspective view of your mechanism, click the **Perspective** button. This command controls the appearance of mechanisms in depth as perceived by normal binocular vision.



#### Parallel Projection

To display a parallel view of your mechanism, click the **Parallel Projection** button. This is the default projection.

### 2.6.4 Zoom and Pan

Fedem provides zooming and panning controls for use in the active view - for example, graph views. The following commands can be accessed on the *Zoom and Pan* tool bar (or *View* menu):



**NOTE:** Some of these commands cannot be used in all views. When commands cannot be used in the current view, their buttons become unavailable (grayed out) on the menus and tool bars.



### Zoom All

To scale the active view so that all objects (for graph views, every curve on the graph) fit within the view, press the **Zoom All** button. When working in graph views, this can also be achieved by pressing the **F5** key.



### Zoom To

This command pops up the correct view, zooms to the selected object, and places the *Dynamic Center* of rotation at the center of the object. This is very useful when trying to locate a certain triad or joint in a large model. This command is also available from the *Topology* view and the Model Manager panel (see [Section 2.4.3, "Model Manager"](#) and [Section 2.4.4, "ID and Topology panel"](#)). It is also applicable on Control Elements and Control Lines in the *Control Editor* view (see [Chapter 7, "Control Editor"](#)).



### Zoom Window

To enlarge a rectangular area, press the **Zoom Window** button. The command can also be activated by pressing the **Z** key.



### Zoom Window With Auto scale

To enlarge a rectangular area, press the **Zoom Window With Auto scale** button. The contents will be auto scaled to fit the entire plotting area. This command can also be activated by pressing the **X** key.



### Zoom In

To enlarge the active view by a predefined scale factor, press the **Zoom In** button.



### Zoom Out

To reduce the active view by a predefined scale factor, press the **Zoom Out** button.



### Pan Left

To move the active view to the left, press the **Pan Left** button.

**Pan Right**

To move the active view to the right, press the **Pan Right** button.

**Pan Up**

To move the active view up, press the **Pan Up** button.

**Pan Down**

To move the active view down, press the **Pan Down** button.

### 2.6.5 General Appearance

The **General Appearance** command can be used to control which entities are displayed in the *Modeler* view. This command also provides control of the size and appearance of mechanism symbols.



Click the **General Appearance** button to open the associated dialog box (shown below).

**① Mechanism symbols** – allows you to toggle on/off the display of mechanism items of different type, to edit the color used to specify each item type, and to change the size of symbols and line widths (see "[Mechanism symbols](#)" below).

**② Default colors** – controls the colors used for Grounded triads, unattached mechanism items and the *Modeler* background (see "[Default colors](#)" below).

**③ Viewer options** – controls 3D rendering options such as visibility, transparency type, and line-smoothing (see "[Viewer options](#)" below).

### Mechanism symbols

This area provides the following controls:

- **Visible** – Enables/disables the display of each item type. Click the box next to an item type to change the setting. The toggles that are indented slightly to the right are considered as sub-types of the item toggle just above them. They are active only if that toggle is enabled.

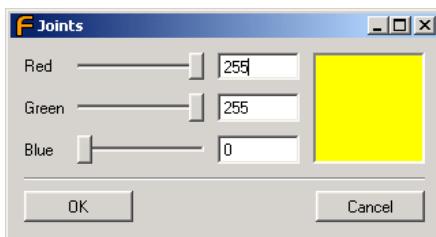


**TIP:** Turning off items speeds up and simplifies the display of complex mechanisms, and provides a useful way to limit selection to specific items.

The screenshot shows the 'General Appearance' dialog box with three main sections:

- ① Mechanism symbols:** A list of mechanism item types with checkboxes and color swatches. Checked items include Revolute joints, Ball joints, Rigid joints, Free joints, Prismatic joints, Cylindric joints, Cam joints, Springs/Dampers, Gears, Loads, Sensors, Strain rosettes, Stickers, and Triads. Sub-items like 'Triads between beam elements' are also listed. There are 'Edit' buttons for each color swatch.
- ② Default colors:** Options for Grounded triads (blue), Unattached symbols (white), and Modeler background (dark blue).
- ③ Viewer options:** Options for Fog (unchecked), Visibility (sliding bar at 0.1), and Simple transparency (unchecked). A 'Close' button is at the bottom.

- **Color** – Allows editing of the RGB settings for each item type. Press the **Edit** button next to an item type to edit the default display color for that item. In the appearing dialog box (shown at right), move the sliders to change the settings or enter the desired values directly in the number fields.
- **Size** – A scale factor for sizing the display of mechanism entities. To change the size of items, enter a new number in the size field.
- **Line Width** – This is a scale factor for the line-width of all mechanism symbol lines, together with all 1D elements and *Surface Connectors* in the FE parts (see [Section 4.6.2, "Surface Connectors"](#)). To change the line-width, move the slider right or left.



### Default colors

This area enables the user to edit the colors used on the modeling background and unattached mechanism items. It also allows you to set the colors on triads that are attached to the ground, to distinguish them from triads that are free to move. The default colors may be changed in a similar ways as changing the colors for [Mechanism symbols](#).

### Viewer options

This area enables the user to modify the way that models are rendered.

- **Fog** is an option that enables you to create a fog-like effect around your model that appears as fog or darkness (or even an underwater scene). The distant parts of the model appear to fade into the background. The Visibility slider controls the distance at which the model is completely hidden in the “Fog”.



**TIP:** You can create the following effects with the Fog option:

- To create the effect of a foggy day, set the Modeler background color to light gray (Red 180, Green 180, Blue 180), enable the **Fog** option, and adjust the **Visibility** slider until the model almost fades into the background.
- To create the effect of an underwater scene, set the Modeler background color to sea green (Red 20, Green 125, Blue 130), enable the **Fog** option, and adjust the **Visibility** slider until the model nearly fades into the background.
- **Simple Transparency** is a dithering algorithm used to speed graphic performance when displaying transparent objects in your model. The effects of this option depend on the type of graphics card you have.

- Anti-Aliasing enables/disables line-smoothing for symbols.



**TIP:** If Anti-Aliasing does not function properly, try enabling the **Simple Transparency** option.



**NOTE:** Graphics cards do not all have the same optimal settings. In general, disabling the **Fog** and **Anti-Aliasing** options and turning on **Simple Transparency** gives the best performance, but with some systems the performance gain is insignificant.

## 2.6.6 Item Appearance

The **Item Appearance** command can be used to change the level of detail and the appearance of individual part and the Reference Plane.



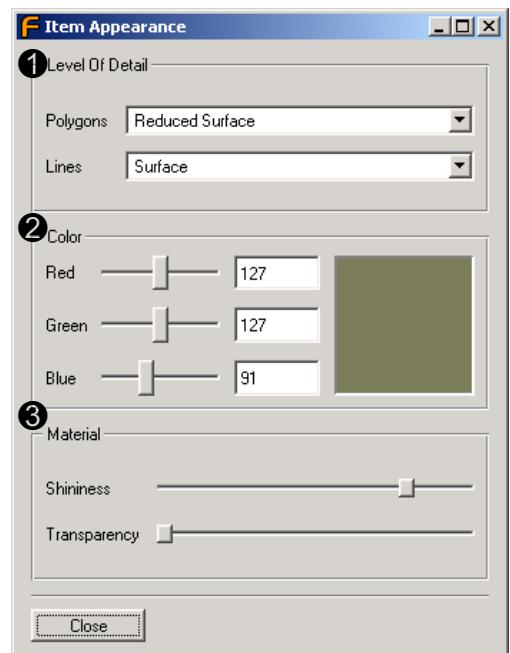
To open the Item Appearance dialog box, click the **Item Appearance** button on the *Standard* tool bar, and select a *Part* or the *Reference Plane*. The Item Appearance dialog box is displayed as shown below.



**TIP:** To change the appearance of a hidden part, select the part in the Objects list of the Model Manager panel, after clicking the **Item Appearance** button.

- ① *Level of Detail* – controls the level of complexity displayed in the model.
- ② *Color* – controls the RGB settings of the selected part or Reference Plane.
- ③ *Material* – controls the shininess and transparency of the selected part or Reference Plane.

In the *Level of Detail* area, the *Polygons* and *Lines* settings allow you to change the complexity of models displayed in the *Modeler* view. Changing these settings can improve the graphic performance of 3D rendering.



*Polygons* can be displayed at five levels of detail: *Surface and Internals*, *Reduced Surface and Internals*, *Surface*, *Reduced Surface*, and *Off*. The default level is *Surface*.

1. *Surface and Internals* - With polygon detail set to Surface and Internals, element faces from solid elements inside the parts are shown together with the surface faces of the elements. All element faces are shown as single polygons.
2. *Reduced Surface and Internals* - Setting the detail level to Reduced Surface and Internals displays a simplified polygon representation of the surface and internals of a part. This is faster than using *Surface and Internals*, but shows a less accurate representation of the parts.
3. *Surface* - This option turns off the internal faces in a part, and will only show the surface element faces of a part. All the surface faces are shown as separate polygons.
4. *Reduced Surface* - Setting the polygon detail to Reduced Surface provides the most efficient way to visualize the shaded view of a mechanism part. The surface is displayed using a simplified polygon model and the internal faces are turned off.
5. *Off* - Setting the polygon detail to Off turns all polygons off.

Lines can be displayed at six levels of detail: *Full*, *Surface*, *Outline*, *Outline No 1D-elements*, *Simplified*, and *Off*. The default level is *Outline*.

1. *Full* - With line detail set to Full, mesh lines from solid elements inside the parts are shown together with the surface mesh of the elements.
2. *Surface* - Setting the line detail level to Surface displays only the mesh lines on the surface of the FE model.
3. *Outline* - This option leaves only the mesh lines on the surface of the part with neighboring element faces with a relative face-angle above a certain threshold. The default threshold is  $\pi/4$ . (It is possible to edit this value for each part by editing the model file.)
4. *Outline No 1D-elements* - Same as Outline except that all *Surface Connectors* and 1-D elements such as RGD, WAVGM, BEAM2 are also removed from the display.
5. *Simplified* - This option generates a simple line visualization of the part based on the Triads attached to it. One line is drawn from each Triad to their geometrical center. This option makes most sense if the polygons are turned off. This visualization is the same that is used if the part is not loaded, see [Section 2.8.1, "FE-Data Settings"](#).
6. *Off* - Setting the line detail to Off turns all mesh lines off.



**TIP:** To edit the appearance or level of detail on several parts at the same time, select multiple parts in the Objects list in the Model Manager panel, after clicking on the **Item Appearance** icon.

## 2.6.7 Element face visibility

The visibility of elements can be controlled using the **Hide Element Faces** and **Show Element Faces** commands in the right-click menu in the *Objects* list of the Model Manager panel. These commands can be applied to the entire part, or to a selection of the element groups listed in the *Objects* list. The icon in front of each group entry in the list indicates the current visual state of the elements in that group: Either all visible , some visible  or all hidden .

These commands will only be active when the polygon detail level is set to *Surface* or *Surface and Internals*, or when color contour results are loaded for the part.

This feature can also be utilized to load color contour results only for small parts of a big part because the color values will not be loaded on hidden elements. See also [Section 9.5.4, "Performance of animation loading"](#).



**NOTE:** The visibility of the mesh is not affected by the **Show/Hide** commands.

## 2.6.8 Visualization of special elements

There are several element types that are treated differently from normal elements like shells and solids, when it comes to visualization. Those elements are listed in the table below.

**Table 2-1: Line style of special elements**

Element type	Visualization	Comments
Beams	Dash dot lines	Any eccentricity is shown as dotted lines from the nodal point to the beam end
Rigid bars	Dashed lines	
Constraint elements (RBE3, WAVGM, Distributed coupling)	Dotted lines	
Concentrated mass	No visualization	
Springs	No visualization	
Bush elements	No visualization	

## Color

The color of those elements are set automatically to black, white or a grayish color to achieve a good contrast to both the color of the part, and the viewer background. If the FE part is displayed using lines only, the color of those elements is set to the color of all the other lines.

## Line width

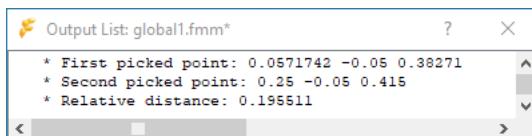
The line width is adjusted according to the Line Width parameter set for the Mechanism symbols. See [Section 2.6.5, "General Appearance"](#).

### 2.6.9 Measuring distance and angles in a model

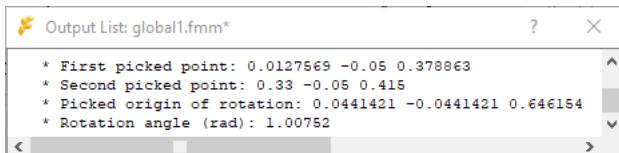
Sometimes during the modeling of complex mechanisms, it is necessary or convenient to quickly assess the distance between two arbitrary points in the model or to measure the angle between two lines. For this purpose, two command are available from the *Tools* menu.



Select **Measure distance...** to check the distance between two points. Then click on two points in the *Modeler* view to output their global positions and the relative distance between them in the *Output List* view. You can repeat to click on two new points to check their relative distance as many times you like. Press **Cancel** to quit the command.



Select **Measure angle...** to check the angle between two lines defined through three selected points. The end points of the two lines are selected first by clicking on two arbitrary locations in the *Modeler* view, followed by the common start point of the two lines. The global positions of the three selected points, as well as the angle (in radians) rotating the line though the third and first point into the line through the third and second are then shown in the *Output List* view. You can repeat to click on three new points to measure more angles as many times you like. Press **Cancel** to quit the command.



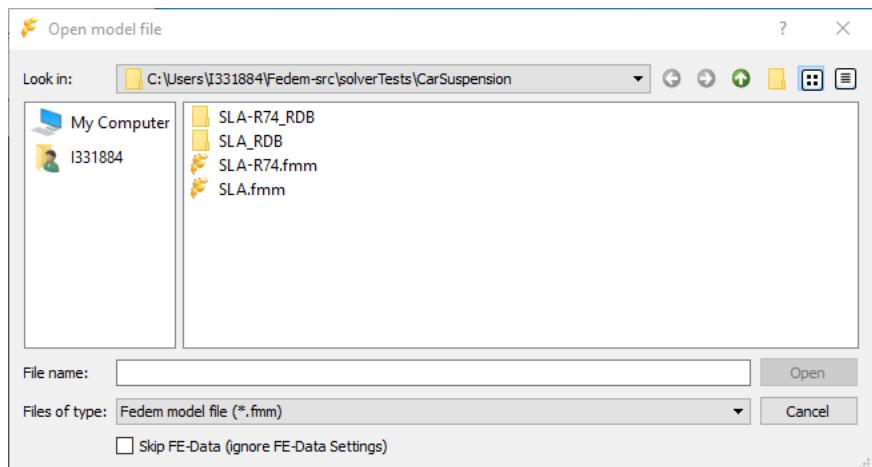
## 2.7 Opening and saving model files

### 2.7.1 Opening a file

You can open a Fedem model file created in this version (or any of the previous versions) of Fedem through the following steps:



1. Choose **Open...** in the *File* menu, or use the **Open** icon in the tool bar, to make the Open model file dialog box appear (shown below).
2. Locate the wanted Fedem model in the dialog box and click **Open**.



2

The Open model file dialog box normally displays all files with the Fedem model file extension (`.fmm`). However, you may open a file with any extension by selecting the *All files* filter in the *File of type* pull-down menu.



**CAUTION:** If you choose to open a file without the `.fmm` extension you should make sure it is a proper Fedem model file. Attempting to open a non-model file will usually result in an empty model, but unpredictable behavior may also occur, depending on the actual contents of the file.

You can skip the loading of FE data for the parts if the model contains large parts that take a long time to load. See also [Section 2.8.2, "Skipping FE-Data when opening a model file"](#).

If the opened model file contains results, information about the result files in the model (`.frs` files) is reported in the *Output List* view. However, results files belonging to unloaded parts (see [Section 2.8, "Loading and unloading FE-Data"](#)) and disabled result files (see [Section 10.2.2, "Result manipulation"](#)) are not included in this report.

If any problems are encountered during model loading, Fedem displays a message box and provides additional information in the [Output List](#) view.



**NOTE:** *There might be minor changes in the model file format from one Fedem version to subsequent versions of Fedem. The newer versions are always backward compatible such that you may safely open a model that was created in one particular version in any of the subsequent versions, without losing model consistency. The model is automatically converted to the new format while reading it.*



**CAUTION:** *The Fedem model files are not necessarily forward compatible. If you open a model in an older version of Fedem than it was created in, there might be changes in the model file format that makes the imported model incorrect or inconsistent. In some cases it may also make the Open operation fail or hang. Refer to the Fedem R5.0 Release Notes, Chapter 3, "Notes", for a summary of the forward compatibility issues that might need to be manually resolved in such cases.*

### Loading parts

When Fedem opens a model file and the loading of FE data is enabled, the part information (.fetl files, reduced matrix files, etc.) is read from the FE model repository, see [Section 5.1.6, "Using FE model repositories"](#).

If the FE model repository is missing, for example when a model file is moved, Fedem uses the following search path to locate the parts:

1. The name and location of the originally imported FE part.
2. The name of the original FE part, located in a sub-directory of the current model file directory, with the same name as in the original FE file path (only if the original path was an absolute path).
3. The name of the original FE part, located in a parallel directory of the current model file directory, with the same name as in the original FE file path (only if the original path was an absolute path).
4. The name of the original FE part, located in the same directory as the model file.
5. The base name of the original FE part with the extension .fetl located in the same directory as the model file.

## 2.7.2 Saving models

To save the current model, do one of the following:



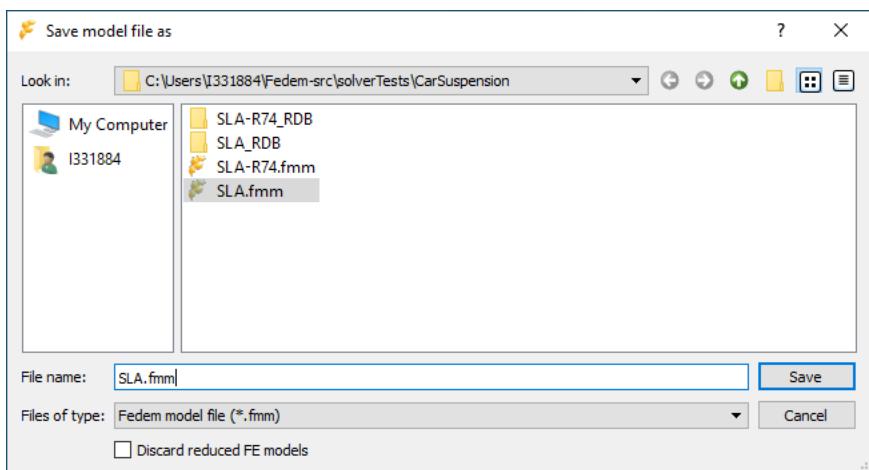
- To replace the current version on disk, choose **Save** in the *File* menu or click on the **Save** icon in the tool bar.



**NOTE:** The previous version of the model file will be renamed to <filename>.bak before writing the new file, such that you can always go back to the previously saved version by renaming that file, in case the last save operation failed due to full disk or other reasons.

- To save the file in a different location and/or with a different name, choose **Save As...** in the *File* menu.

When saving a copy of the model using the **Save As...** command, a dialog box labeled "Save model file as" appears (shown below), in which you can choose the new file name. Here you can also choose to **Discard reduced FE models** by enabling the associated toggle (this toggle is present only when the model contains FE parts).



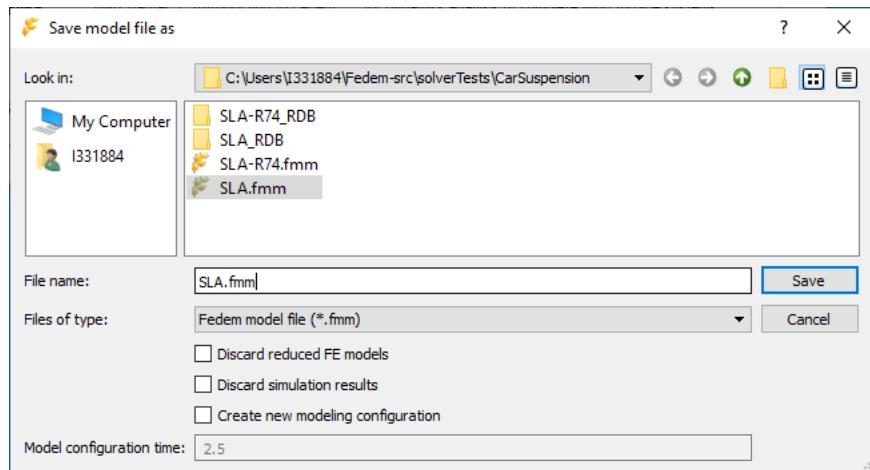
When saving a new model for the first time, you are prompted to give it a name different from the default name *untitled\_<#>.fmm*, which was assigned when Fedem was started (see [Section 2.3, "Starting Fedem"](#)). If you also have performed some solver tasks before saving the model, the existing results database will then be moved to the correct location associated with the new model file name, unless you have toggled on either **Discard simulation results** or **Create new modeling configuration**, see "[Saving a model copy with results](#)" below.



**CAUTION:** If you do not save a new model still named *untitled\_<#>.fmm* before you open another model or **Exit** Fedem, all solver results associated with this model will be deleted, if any. This also includes the results of any reductions performed, unless FE model repositories were used (see [Section 5.1.6, "Using FE model repositories"](#)).

### Saving a model copy with results

If the model contains simulation results when you do a **Save As...**, you will have the options to either **Discard simulation results** or **Create new modeling configuration**, by activating the associated toggles:



If the latter toggle is activated, you can then specify the simulation time from which the response state will be used to create a new modeling configuration, by updating all Triad and Part positions in the new model.

#### Indication on whether a **Save** is needed

When the current model has been changed compared to the previously saved version, Fedem will indicate this with an asterisk (\*) after the model file name in the title bar of the main window. Only if this asterisk is present, you will be asked if you first want to save the current model when you **Open** another model, create a **New** model, or **Exit**.



### 2.7.3 Starting a new model



You can at any time start modeling on a completely new model in the current Fedem session by choosing **New** from the *File* menu, or by clicking the **New** icon in the tool bar. This is equivalent to exiting the current Fedem session and then starting a new one (see [Section 2.3, "Starting Fedem"](#)) except that the new model now by default will be located in the same directory as the current model.



**NOTE:** If the current model has been modified after the last **Save**, you will be asked whether you want to save those changes before starting on the new model.

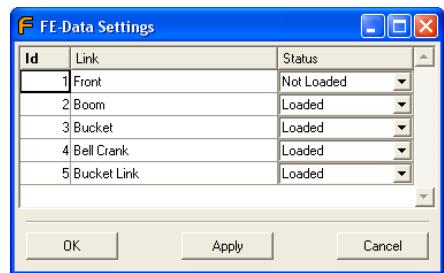
## 2.8 Loading and unloading FE-Data

When a model contains several large FE-models, unloading some of the FE-data from memory can be necessary to in order to reduce the amount of resources used. This is particularly useful when you want to load contour plots for a particular part, or to free up resources for the solvers.

### 2.8.1 FE-Data Settings



FE models often tend to be large. The amount of data needed for visualization and lookup is indeed significant. In some cases it will be convenient or necessary to unload this data, to free up RAM. To control the loading and unloading of FE-data, the FE-Data Settings dialog box is used. To open this dialog box, select the **FE-Data Settings...** command in the *Tools* menu.



The status of each part is set using the pull-down menu in the *Status* column. Set the status you want for each part, and press **OK** or **Apply**.

The parts that are not loaded will be shown using the *Simplified* line shape, as described in [Section 2.6.6, "Item Appearance"](#).

The loading and unloading of parts does not affect the simulation. The solver processes will read the necessary FE-Data from the FE-model files.



**NOTE:** Loading unloaded parts can take several minutes if the parts are big.



**TIP:** To change the status of all or several parts quickly, it is convenient to use the arrow keys to navigate between the pull-down menus, and the **N**-key (Not Loaded) or **L**-key (Loaded) to set the status.

### 2.8.2 Skipping FE-Data when opening a model file

When opening a big model for inspection or simulation, it is sometimes convenient to override the FE-Data Settings and skip the FE-models completely. In that way you can open a model containing extremely large parts in a few seconds, without using any significant amount of memory.

To do this, toggle the **Skip FE-Data** toggle in the Open model file dialog box when opening a model file, or use the `-noFEData` command-line option together with `-f <modelfilename>` when starting Fedem.

The model will then be loaded with all FE parts unloaded. To load the parts, open the FE-Data Settings dialog box which now will show the status on the parts last time you saved your model. If you simply press **OK** or **Apply**, those settings will be applied and the parts that was marked as **Loaded** will then actually be loaded.

### 2.8.3 Modeling with unloaded parts

An unloaded part can generally be used as any part while modeling, but there are some restrictions.

When selecting points on the part, the points will not snap to the closest node, because the node information is not loaded. The only points on the part that are available for selecting and modeling are the external nodes represented by the triads.

Triads can generally not be detached or deleted from the unloaded part. This is done to protect the reduced matrices from being accidentally invalidated. If you try to remove or detach a triad on an unloaded part you will get an error. When detaching joints from an unloaded part, a triad will be left on the part where the joint was attached.

### 2.8.4 Postprocessing unloaded parts

Unloaded parts will be completely skipped when loading an animation, except for the rigid body motion. This makes it possible to focus the computer resources on the parts of your model that are interesting, and skip everything else, see also "*Disabling and Enabling results*" in [Section 10.2.2](#).

## 2.9 Exporting objects

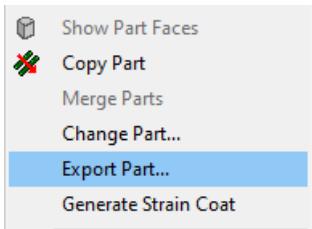
Fedem can export six types of objects: *Parts* (FE models), individual *Curves*, *Graphs*, *Graph* views, the *Modeler* view and *Animations*.

- *FE Parts* can be exported in Fedem Technology Link (`.ftl`) format.
- *Curves* and *Graphs* can be exported in ASCII (`.asc`, `.txt`), nCode DAC (`.dac`) or MTS RPC time history (`.rsp`, `.drv`, `.tim`) format.
- The *Modeler*- and *Graph* views can be exported as binary image files in a variety of formats.
- *Animations* can be exported as movies in mpeg-1, mpeg-2, and avi formats.

## 2.9.1 Exporting a part

To export an FE Part, complete the following steps:

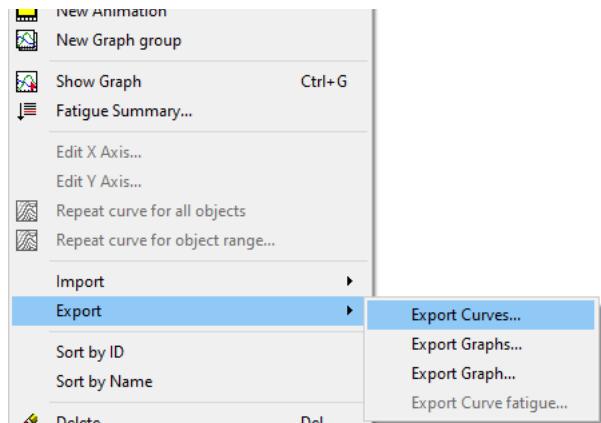
1. Right-click the part in the Model Manager *Objects* list to access the shortcut menu (partly shown right).
2. Select **Export Part...** to open the dialog box labeled *Save part as*.
3. Specify a file name and location, then click **Save**.



## 2.9.2 Exporting curves and graphs

To export curves, complete the following steps:

1. Select one or more curves in the Model Manager *Results* list, and right-click your mouse to access the shortcut menu (shown right).
2. Select **Export Curves...** to open the dialog box labeled either *Save curve as* or *Save curves to directory*.
3. Depending on your selection, you will be prompted for either a *File name* or a *Directory*. If you selected multiple curves, the curve files will be named automatically, but you have to select what *File format* you want to export to.



To export one or more graphs, complete the following steps:

1. Select one or more graphs or curves in the Model Manager *Results* list and right-click your mouse to access the shortcut menu.
2. Select **Export** and then **Export Graphs...** to open the dialog box labeled either *Save graph as* or *Save graph to directory*.
3. As when exporting curves, you will be prompted for either a *File name* or a *Directory* depending on your selection. The graph files will be given names automatically if you have selected multiple graphs.

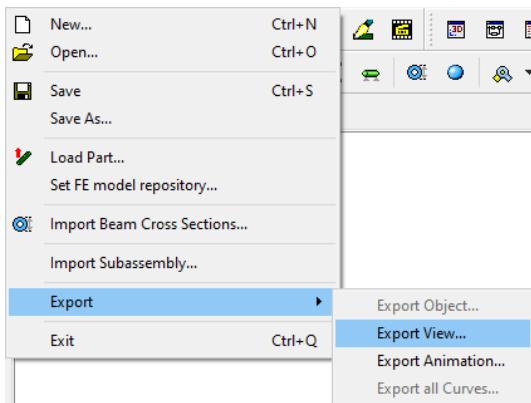
For more detailed information on how to export and import curves and graphs, see [Section 9.2.11, "Export of Curve Data"](#) and [Section 9.2.12, "Importing Curves and Graphs"](#), respectively.

### 2.9.3 Exporting the 3D modeler view

To export this view, make it active, then select **Export** and then **Export View...** from the **File** menu.

When exporting the *Modeler* view, you may choose between the following image formats:

- bmp
- jpeg
- png
- rgb
- 3D inventor snapshot, iv



The different file formats have different quality. *Jpeg* is a widely recognized format. The compression reduces the quality somewhat, but the files are small. *Png* and *bmp* have better quality. We recommend using the *png* format for high quality images.

The *iv* format enables dynamic 3D-viewing of your models using an external viewer. Viewers are available for several platforms.

Stills of animations, e.g., contour plots, can also be exported. Simply pause your animation where you want the picture or 3D-snapshot taken, then export the *Modeler* view.

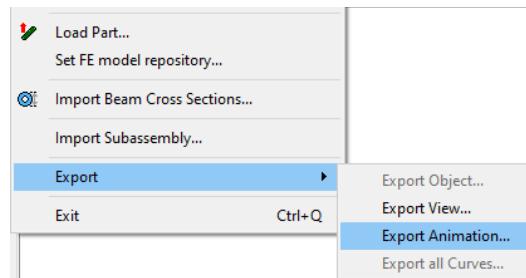
### 2.9.4 Exporting graph views

To export a graph view, follow the steps described in the section above. Here you may choose between these image formats:

- bmp
- jpeg
- png

## 2.9.5 Exporting animations

Animations can be exported using the *mpeg-1*, *mpeg-2* and *avi* (Windows only) formats. The exported animation may be viewed in any standard video player, e.g., Windows Media Player or Elecard MPEG Player ([www.elecard.com](http://www.elecard.com)).



After loading the animation, select **Export** and then **Export Animation...** from the *File* menu to open a dialog box where you can select location, file name and format of your exported animation. See [Section 9.6.4, "Exporting animations"](#) for details.

## 2.10 License information

The Fedem software consists of a foundation module and several add-on modules. The actually installed and available modules depend on your license contract with SAP SE.

### 2.10.1 Available modules

The currently installed licenses are listed to the console window when Fedem is started, if the command-line option *-licenseinfo* is specified. The list may also be shown in the [Output List](#) at any time during a Fedem session by selecting the **License Information** command in the *Help* menu. The list indicates the licenses that are required upon startup, and the add-on licenses that are available to the user:

*License information:*

Module	Name	Version
FF-MDC	Modeler Core	4.00
<i>Available, but not checked out:</i>		
FA-CTI	Control interface add on	4.00
FA-CTR	Control add on	4.00
FA-DRB	Durability add on	4.00
<i>License file/server:</i>		
27000@devpc7 (devpc7)		

For example, when loading a model that has control elements, the control interface add-on license (FA-CTI) is also checked out. This license is held by the application until the model file is closed again.

The license information list is available at any time from the **License information...** command in the *Help* menu.

### 2.10.2 License denial

If the requested add-on license is unavailable (either in use by another session, or not a part of the license contract), modeling, manipulation, and solving of objects covered by that license is denied (e.g., control block editing is not allowed if the control interface add-on is missing). Post-processing (including graph and animation handling) is always allowed as long as the modeler core license is available.

### 2.10.3 License file/server

The current license file/server used is also shown in the license information listing (see [Section 2.10.1, "Available modules"](#)).

### 2.10.4 Managing license files/servers

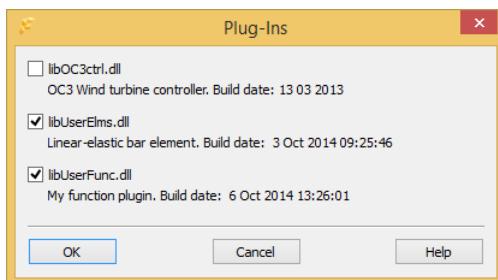
By selecting the **License Manager...** command from the *Help* menu, you can choose which license file to use (by using the **Browse...** button), or edit the name of the license server if you have a floating license.



**CAUTION:** To change license server or path to the license file, you have to run Fedem in Administrator mode on Windows 7 and later, in order for the new settings to be saved. Otherwise, you will have to repeat the license settings each time you start Fedem.

## 2.11 Customizing Fedem using plug-ins

On Windows, plug-ins are provided as DLL-files in the “plugins” subfolder of the installation folder of the Fedem software. Once a plug-in DLL has been placed in the “plugins” folder, it will show up in the Plug-Ins dialog box in Fedem (see below), which is available from *Tools* menu. Here you can tick which plug-ins to use.



Fedem currently provides plug-in interfaces of two kinds; user-defined functions and user-defined elements. However, only one plug-in of each kind can be activated at the same time.

### 2.11.1 User-defined functions

Fedem provides a range of built-in mathematical functions that may be used to describe external loads, prescribed motions, etc., in the model (see [Section 5.11, "Functions"](#)). In addition, a plug-in architecture for enabling user-defined functions is provided.

If you wish to implement your own custom functions, you can create your own plug-in DLL. In the “Templates” subfolder of your Fedem installation, you will find the source code file *userfunc.C*, which can be used as a template for building your own plug-in with functions. Build-instructions and other explanations can be found as comments in the file.

The user-defined functions from the DLL library will show up in Fedem similarly as the internal functions described in [Section 5.11, "Functions"](#).

A user-defined functions plug-in must contain all of the following C-functions. The core functionality of the plug-in lies in evaluation of the function **ufGetValue**. You are free to implement the body of this function as you like, utilizing any other functions and libraries you have available. This allows for very flexible and powerful additions to the Fedem software, and can be used to create black-box control systems, or for calling 3rd party calculation engines on the fly for each solver iteration.

**bool ufGetSignature (int nchar, char\* sign)**

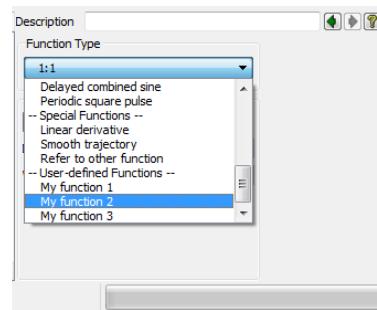
This method returns (through the *sign* argument) the name/signature of the plug-in library, as it is shown in the Plug-Ins dialog box (see above).

**int ufGetFuncs (int maxUF, int\* funcId)**

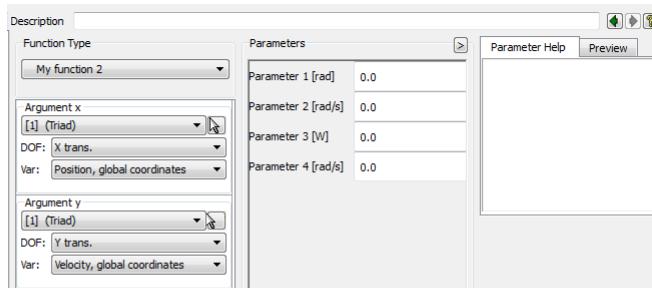
This method returns (through the *funcId* argument) a list of unique IDs for each of the user-defined functions in the library. It can be any integer value, and is used for internal book-keeping only. The method returns the number of user-defined functions in the library.

**int ufGetFuncName (int id, int nchar, char\* name)**

This method defines the *name* of the function with the specified *id*. This is the function name as it appears in the *Function type* menu of the Function Property Editor panel (shown to the right, see also [Section 5.11.2, "Function properties"](#)). The method returns the number of function arguments.

**int ufGetParName (int id, int ipar, int nchar, char\* name)**

This method defines the *name* of function parameter number *ipar* for the function with the specified *id*. This is the parameter name as it is shown in the parameter list of your function in the Function Property Editor panel (shown below). The method returns the total number of parameters for the specified function.



```
double ufGetValue (int baseld, int funcId, const double* par,  
                  const double* x, int& err)
```

This method evaluates an instance of the specified function, for a given set of parameter values (*par*) and argument values (*x*). This is where you will insert your user-specific code that performs the function evaluation.

The *baseld* argument is a unique identifier for each function instance, and is assigned automatically by Fedem. It may be used to identify a certain instance (if there are more than one) of a given user-defined function within the plug-in library. The *funcId* argument then has the same value for all instances of a given function type.

```
double ufGetDiff (int baseld, int funcId, int ia, const double*par,  
                  const double* x, int& err)
```

This method evaluates the *ia*<sup>th</sup> partial derivative of the specified function for a given set of parameter and argument values. If you do not require the function derivatives in your application, you can just leave the body of this method blank and return 0.0.



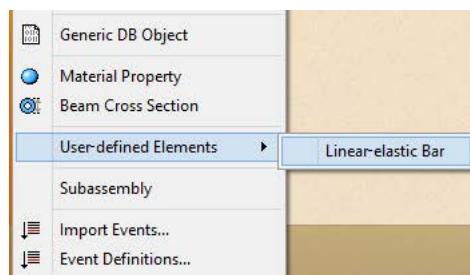
**NOTE:** You should be aware of the difference between arguments and parameters for a user-defined function. Arguments are the variables that are linked to simulation results from Fedem, such as position, angle, velocity, etc., for a particular part of your model. Parameters are additional constant variables that you need to evaluate the function. Say, your function is some mathematical equation, then the arguments are the equation's variables, while the parameters are the equation's various constants.

## 2.11.2 User-defined elements

Fedem now provides a plug-in architecture for enabling user-defined finite elements in a mechanism model.

If you wish to implement your own custom elements, you can create your own plug-in DLL. In the “Templates” subfolder of your Fedem installation, you will find the source code file *userelm.C*, which can be used as a template for building your own plug-in with elements. Build-instructions and other explanations can be found as comments in the file.

When a plug-in DLL with user-defined elements is activated, the element types contained by it will be listed under the *User-defined Elements* sub-menu of the *Mechanism* menu, as shown to the right. In this example, the DLL contains one element type named "Linear-elastic Bar".



A user-defined elements plug-in must contain all of the following C-functions. The core functionality the plug-in lies in the evaluation of the function **ueUpdate**. You are free to implement the body of this function as you like, utilizing any other functions and libraries you have available. This allows for very flexible and powerful additions to the Fedem dynamics solver, providing sets of black-box nonlinear finite elements.

#### **bool ueGetSignature (int nchar, char\* sign)**

This method returns (through the *sign* argument) the name/signature of the plug-in library, as it is shown in the Plug-Ins dialog box.

#### **int ueGetElements (int maxUE, int\* eType)**

This method returns (through the *eType* argument) a list of unique IDs for each of the user-defined element types in the library. The IDs can be any integer value, and is used for internal book-keeping only. The method returns the number of user-defined element types in the library.

#### **int ueGetTypeNames (int eType, int nchar, char\* name)**

This method defines the *name* of the element type with the specified *eType*. This is the element type name as it appears in the *User-defined Elements* sub-menu of the *Mechanism* menu (as shown above). The return value is the number of nodes in the specified element type.

#### **int uelinit (int eld, int eType, int nenod, int nedof, const double\* X0, const double\* T0, int\* iwork, double\* rwork)**

This method initializes a given user-defined element, and is invoked twice by the dynamics solver for each element instance (identified through *eld* and *eType*), before the time integration loop is started. The function arguments have the following interpretation:

1. *eld* - Unique ID identifying this element instance (the base ID)
2. *eType* - Unique ID identifying the element type
3. *nenod* - Number of element nodes
4. *nedof* - Number of element degrees of freedom
5. *X0* - Global nodal coordinates of the element (initial configuration)
6. *T0* - Global nodal orientations of the element (initial configuration)
7. *iwork* - Integer work area for this element
8. *rwork* - Real (double precision) work area for this element

In the first call, the arguments *X0*, *T0* and *rwork* must all be NULL pointers, and the method then only calculates the required size of the work areas *iwork* and *rwork* of that particular element. These size parameters are stored in *iwork[0]* and *iwork[1]*, respectively. The work areas are then allocated by the calling program and the method is called the second time, performing all calculations that are needed to be done only once for each element instance. The results of these calculations are stored in the work arrays *iwork* and *rwork* for later reference by **ueUpdate**.

The method returns zero on successful exit, and a negative value if any exception occurs. In the latter case the program execution is aborted.

```
int ueUpdate (int eld, int eType, int nenod, int nedof, const double* Xn,
              const double* Tn, const double* Vn, const double* An,
              int* iwork, double* rwork, double* eKt, double* eC,
              double* eM, double* eFs, double* eFd, double* eFi,
              double* eQ, double t, double dt, int istep, int iter)
```

This method is invoked by the dynamics solver, once for each element instance within the nonlinear iteration loop. It evaluates the updated tangent matrices and associated right-hand-side force vectors for the specified element. The body of this function is where you will insert your user-specific code describing the non-linear element behavior.

The function arguments no. 5 through 17 (*Xn*, *Tn*, ..., *eQ*) are all C-pointers to arrays which are allocated by the calling program. The length of these arrays depends on the number of nodal points connected to the element, and it is the responsibility of the programmer making the DLL to ensure that access outside the array bounds does not occur. The interpretation of all the 21 function arguments are as follows:

1. *eld* - Unique ID identifying this element instance (the base ID)
2. *eType* - Unique ID identifying the element type

3. *n nod* - Number of element nodes
4. *n edof* - Number of element degrees of freedom
5. *Xn* - Global nodal coordinates of the element (current configuration)
6. *Tn* - Global nodal orientations of the element (current configuration)
7. *Vn* - Global nodal velocities of the element (current configuration)
8. *An* - Global nodal accelerations of the element (current configuration)
9. *iwork* - Integer work area for this element
10. *rwork* - Real (double precision) work area for this element
11. *eKt* - Element tangent stiffness matrix
12. *eC* - Element damping matrix
13. *eM* - Element mass matrix
14. *eFs* - Internal elastic forces of the element
15. *eFd* - Internal damping forces of the element
16. *eFi* - Internal inertia forces of the element
17. *eQ* - External forces acting on the element
18. *t* - Current time
19. *dt* - Time step size
20. *istep* - Time step number
21. *iter* - Nonlinear iteration number

The method returns zero on successful exit, and a negative value if any exception occurs. In the latter case the program execution is aborted.

### **Creating user-defined elements**

When creating user-defined elements in a Fedem model, they have to be created from a set of already created triads, in a similar fashion as when creating Beams and Generic Parts:

1. Create Triads where you want to locate the nodes of the elements.
2. Select the triads representing the nodes of the element(s), using the multi-select features of Fedem (see [Section 2.4.3, "Model Manager"](#) and [Section 2.5.1, "Select"](#)).
3. Select the appropriate item from the *User-defined Elements* sub-menu of the *Mechanism* menu, that represent the type of element you want to create.

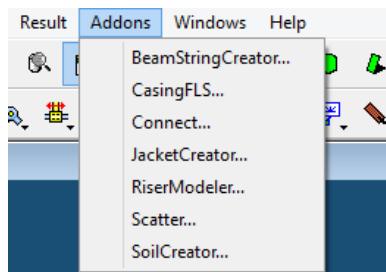
A set of user-defined elements of the specified type is then created. The number of elements created is equal to  $(nT-nEn+1)$  where  $nT$  is the number of selected triads, and  $nEn$  is the number of nodes per element, and two elements will then always have  $(nEn-1)$  nodes in common. This can be utilized to create strings of two-noded user-defined elements.



**CAUTION:** Pay attention to the order in which you select the Triads to be used, as it will reflect the internal nodal ordering of the created element. For elements with three or more nodes, the internal ordering is often of importance to get a proper solution.

## 2.12 Customizing Fedem using Addons

On the Windows platform (Windows 7 and later), Fedem now offers the possibility to extend the modelling and post-processing environment through the use of **Addons**.



The Addons are separate executables that may be launched from the *Addons* menu in Fedem (as shown to the right). Each menu item here corresponds to an EXE file in the “Addons” subfolder of the installation folder of the Fedem software. If that folder is empty or does not exist, the *Addons* menu is absent.

The Addons communicate with the main Fedem GUI over a COM API, which is available for advanced users such that they may program their own addons for performing certain tasks. This is a powerful means for creating specialized pre- and post-processors for doing complex modeling tasks. The Fedem installation does not contain any Addons itself, but some can be provided as additional packages.

Contact support at the *SAP Engineering Center of Excellence* if you wish to explore the Addons capability by programming your own extensions.



# Chapter 3 Wind Power Modeling



This chapter describes how to perform the basic operations you need to do, in order to build wind turbine models, such as defining the turbine, defining the tower, adjusting the model, etc.

Fedem WindPower uses the AeroDyn software module of the National Renewable Energy Laboratory (NREL) in the USA (<http://wind.nrel.gov>), for all calculations related to the aerodynamic loads on the wind turbine blades. These loads are then applied onto the structural model of the wind turbine in the Fedem solvers, to provide response data that can be further analysed.

3

The basic mechanism elements of Fedem (parts, triads, joints, etc.) and their properties are discussed in detail in [Chapter 5, "Mechanism Elements"](#). This chapter focuses on issues specific to wind turbine modeling only.

Sections in this chapter address the following topics:

- [Wind turbine modeling approaches](#)
- [Creating a basic wind turbine model](#)
- [Basic solving and analysis](#)
- [Creating an advanced wind turbine model](#)
- [Advanced solving and analysis](#)

## 3.1 Wind turbine modeling approaches

There are alternative approaches to wind turbine modeling:

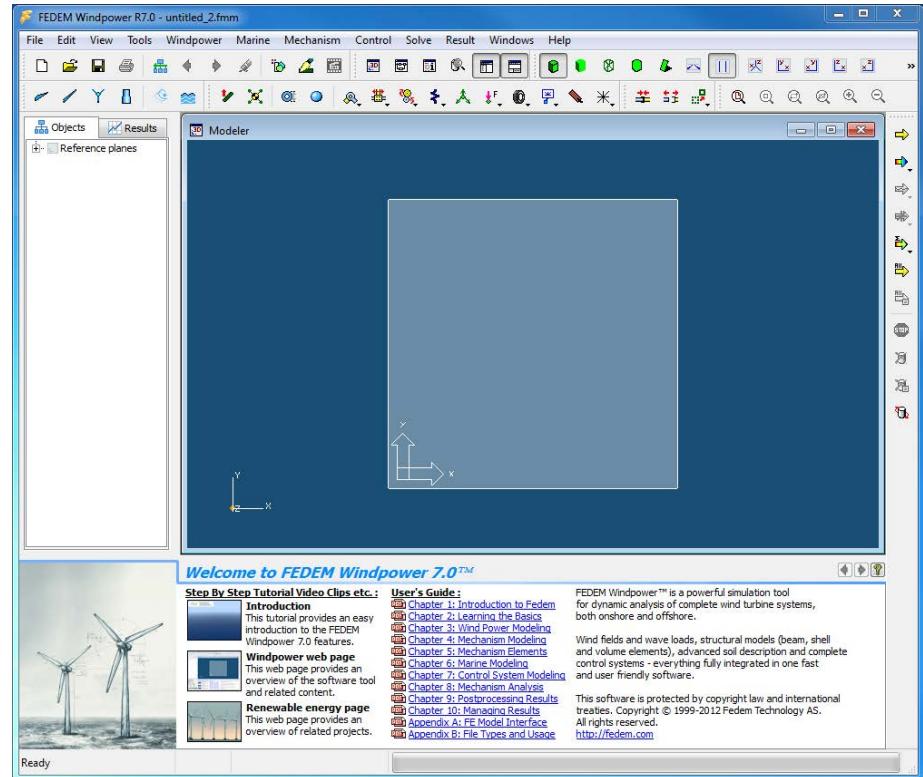
- *NREL/FAST*- The FAST software is the first generation of wind turbine modeling tools. The software was/is developed as a part of a public research project at the National Renewable Energy Laboratory. The FAST software is open source, which is important in many research projects that seek to advance scientific knowledge within the area. The tool does, however, not have a graphical user interface and is difficult to use correctly in a fast-paced commercial application.
- *GLGH/Bladed*- The Bladed software is the first generation of privately developed and commercially available wind turbine modeling tools. The tool basically adds a graphical user interface to the wind turbine modeling process. A set of dialog boxes are used to input options and data. The tool offers a straight-forward approach to the wind turbine modeling process that works well for typical wind turbine models.
- *Fedem WindPower*- We, respectfully, offer Fedem WindPower as a next generation wind turbine modeling environment. The Fedem software is designed and developed to be a high-end technology platform and an engineering framework for virtual testing of complex mechanical assemblies. It provides a complete set of features to create, solve and post-process wind turbine models in a 3D graphical environment. We can offer our software as a complete replacement of prior software.

There are additional wind turbine modeling tools as well that are made at universities, research institutions and elsewhere but they are in the design line of first generation tools, for the most part.

Fedem WindPower is the only software that offers a high-end technology platform and engineering framework for virtual testing of complex wind turbine models. Fedem WindPower includes a large array of mechanism elements such as parts, joints, springs, dampers, loads, control systems, and so on, that can be added to (or modified in) the wind turbine model. Each mechanism element has a large array of properties that offers fine detail control over the model. The solvers produce results that offer an easy and intuitive analysis approach with integrated graphs and tables. Results can also be exported to other tools. Never before have wind turbine modeling, solving and analysis been this easy, flexible, extensible, comprehensible, precise, auditable and powerful!

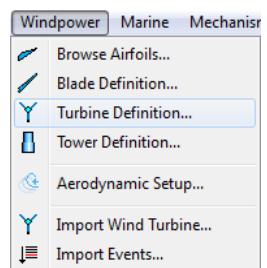
## 3.2 Creating a basic wind turbine model

The Fedem WindPower user interface is illustrated in the figure below. Here we see the start up screen when you have just started the software.

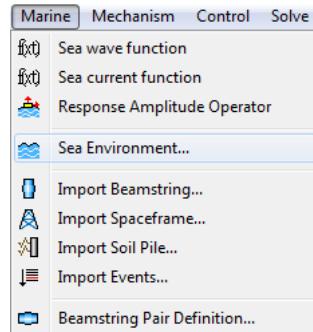


Fedem WindPower has a few major additional features, as compared to the standard edition:

- **Windpower menu** - The main window includes the **Windpower** menu (shown at right). It contains choices for creating and updating the wind turbine model, for aerodynamic setup, etc.
- The **Import Wind Turbine...** entry provides an alternative (obsolescent) way of specifying a turbine setup. It is present for backward compatibility only and should not be used when starting a new project.



- The **Import Events...** entry is used for setting up simulation events, which is handy when you have a large simulation setup with menu load cases on an (almost) identical structural model. Refer to [Section 8.16, "Using simulation events"](#) for learning more on how to set up and use simulation events.
- *Marine menu* - The main window also includes the *Marine* menu (shown at right). It contains choices for defining the marine environment when modeling an offshore wind turbine. Refer to [Chapter 6, "Marine Modeling"](#) for full details on these items. The *Marine* menu also contains entries for importing higher-level structural components that are subjected to marine loads, such as space frames (Jackets), soil piles and general beam strings. It also contains the same *Import Events...* entry as in the *Windpower* menu.
- *Tool bars*- The *Windpower* tool bar (shown at right) provides an easy access to some choices of the *Windpower* and *Marine* menus.
- *Start Guide*- A start guide in the Property Editor panel (shown below) provides links to assistance and documentation. We highly recommend viewing the video tutorials to get a quick and high-level feeling of the software before proceeding.



**Welcome to FEDEM Windpower 7.0™**

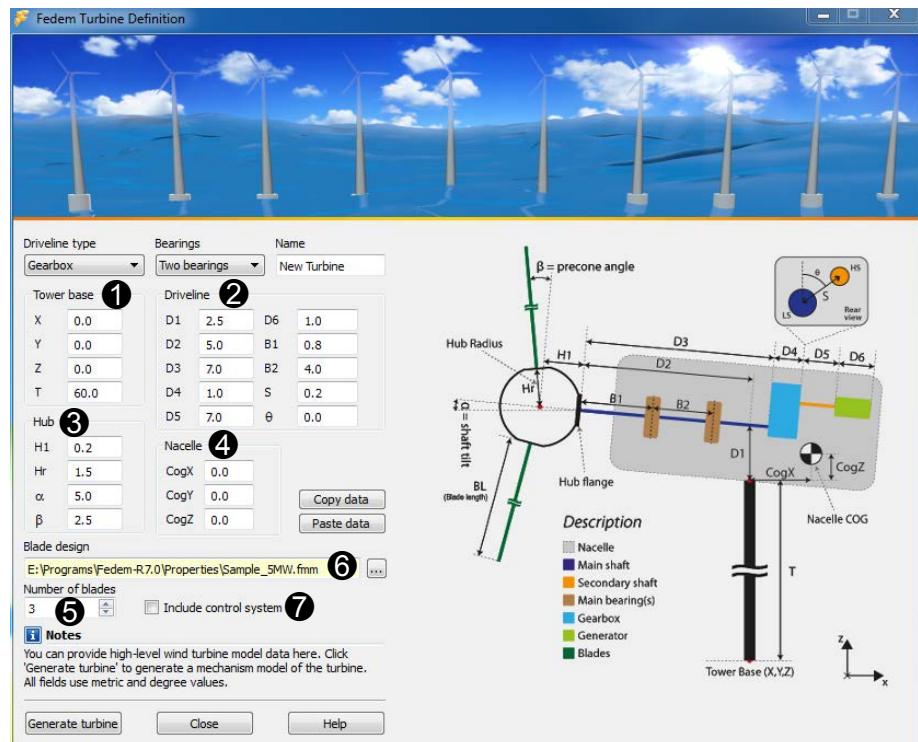
Step By Step Tutorial Video Clips etc.:	User's Guide:	FEDEM Windpower™ is a powerful simulation tool for dynamic analysis of complete wind turbine systems, both onshore and offshore.
 <b>Introduction</b> This tutorial provides an easy introduction to the FEDEM Windpower 7.0 features.	<a href="#">User's Guide</a> : <a href="#">Chapter 1: Introduction to Fedem</a> <a href="#">Chapter 2: Learning the Basics</a> <a href="#">Chapter 3: Wind Power Modeling</a> <a href="#">Chapter 4: Mechanism Modeling</a> <a href="#">Chapter 5: Mechanism Elements</a> <a href="#">Chapter 6: Marine Modeling</a> <a href="#">Chapter 7: Control System Modeling</a> <a href="#">Chapter 8: Mechanism Analysis</a> <a href="#">Chapter 9: Postprocessing Results</a> <a href="#">Chapter 10: Managing Results</a> <a href="#">Appendix A: FE Model Interface</a> <a href="#">Appendix B: File Types and Usage</a>	Wind fields and wave loads, structural models (beam, shell and volume elements), advanced soil description and complete control systems - everything fully integrated in one fast and user friendly software.
 <b>Windpower web page</b> This web page provides an overview of the software tool and related content.		
 <b>Renewable energy page</b> This web page provides an overview of related projects.		This software is protected by copyright law and international treaties. Copyright © 1999-2012 Fedem Technology AS. All rights reserved. <a href="http://fedem.com">http://fedem.com</a>

### 3.2.1 Turbine definition

Fedem WindPower provides a set of dialog boxes for creating a wind turbine model. Let's first have a look at the Turbine Definition dialog box.

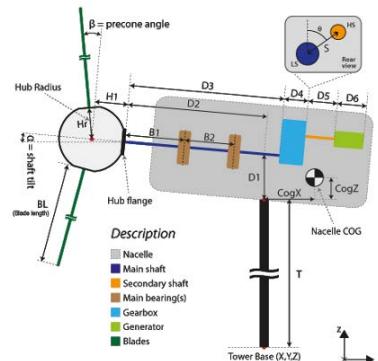


To open the Turbine Definition dialog box, click the **Turbine Definition...** button on the menu or tool bar. The Turbine Definition dialog box is shown below with the default turbine assembly settings.

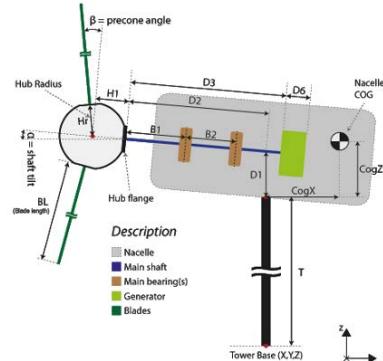


The Turbine Definition dialog box is used to enable easy configuration and creation of wind turbine models. All the fields have predefined default values, so the only thing you need to do in order to generate a valid wind turbine model is to click the **Generate turbine** button.

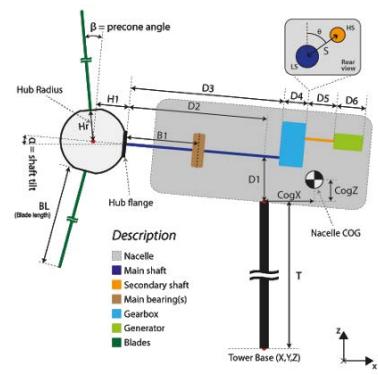
However, before that you may choose another *Driveline type*, the number of *Bearings*, the *Number of blades* as well as selecting a pre-defined *Blade design*. You may also specify a *Name* for the turbine. The figures on the next page illustrate the different driveline and bearing configurations that are currently supported.



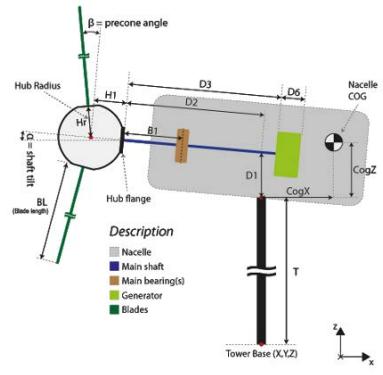
Gearbox, two bearings



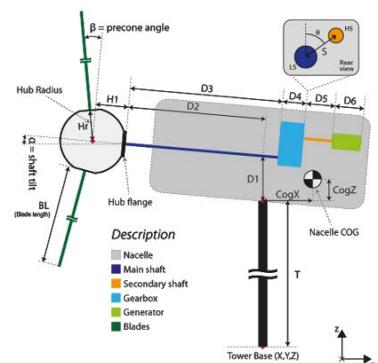
Direct, two bearings



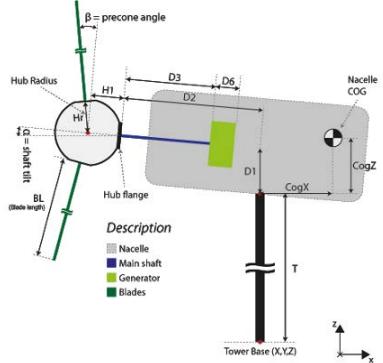
Gearbox, one bearing



Direct, one bearing



Gearbox, no bearings



Direct, no bearings

The various fields in the Turbine Definition dialog box are as follows:

- ❶ *Tower base* - These fields specify the turbine position and the height of the turbine tower:
  - X, Y, Z - Global position of the bottom of the tower.
  - T - Height of the tower. Specify the value 0.0 to generate a turbine model without a separate Tower object.
- ❷ *Driveline* - These fields specify the location and size of various items inside the turbine nacelle:
  - D1 - Distance from the tower top to the main shaft.
  - D2 - Distance from the tower top to the hub flange.
  - D3 - Distance from gearbox to the hub flange.  
This also equals the length of the main shaft.
  - D4 - Depth of the gearbox.
  - D5 - Length of the secondary shaft.
  - D6 - Depth of the generator.
  - B1 - Distance from the hub flange to the first bearing.
  - B2 - Distance between the two bearings.
  - S - Offset distance between the shaft axes.
  - $\theta$  - Offset angle of the secondary shaft.
- ❸ *Hub* - These fields specify the positions and angles of the hub:
  - H1 - Distance from the hub flange to the hub apex.
  - Hr - Radius of the hub.
  - $\alpha$  - Uptilt angle of the main shaft.
  - $\beta$  - Precone angle of the blades.
- ❹ *Nacelle* - These fields provide information about the nacelle:
  - CogX, CogY, CogZ - Position of the center of gravity for the nacelle, relative to the top of the tower.
- ❺ *Number of blades* - Specifies the number of blades for the turbine. Legal values are 2, 3 and 4.
- ❻ *Blade design* - Specifies the blade design to use. A set of sample blades are included with the Fedem WindPower installation. They are stored as separate model files and can be selected using the browse button [...] to the right of the blade design field. You can also make your own blade designs by using the Blade Definition dialog box, as shown in [Section 3.4.2, "Blade definition"](#).



**NOTE:** The color of the Blade design field will be pink as long no valid path to a blade design file has been specified. It will change to yellow once a valid file path is entered. A green field means that this blade design is now included in the current model.

- 7 **Include control system**- If this toggle is enabled, a pre-defined control system for controlling the pitching and torque of the turbine will be generated, see [Section 3.4.5, "Control system"](#).

The **Generate turbine** button will generate a mechanism assembly of the defined turbine, consisting of beam elements, triads, joints, etc. However, if turbine a model already has been created, the same button is labeled **Update turbine** and the positions and properties of the existing turbine components are then updated according the field values.



**NOTE:** If a turbine model already exists, some of the fields in the dialog box cannot be changed and are therefore disabled (gray). These are Driveline type, Bearings and Number of blades. You must delete the existing model before these can be changed.

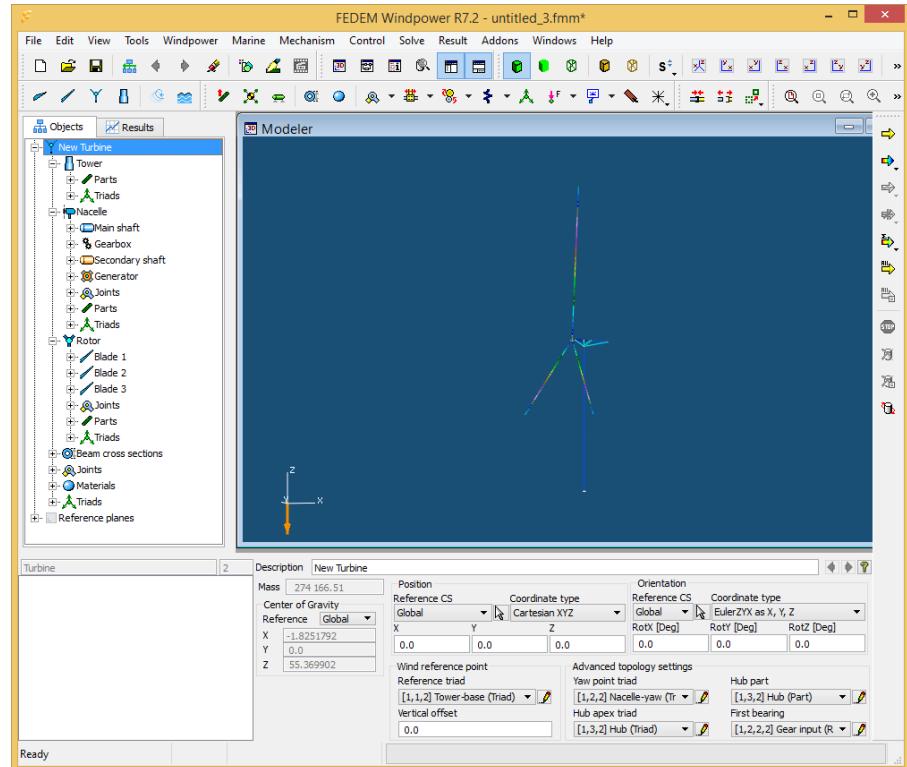
The Turbine Definition dialog box also provides a **Copy data** button and a **Paste data** button to simplify the process of experimenting with many different turbine designs. Simply click the **Copy data** button to copy the fields of the dialog box to the clipboard. You can paste the data back into the dialog box by clicking **Paste data**, or paste the data to other applications. The paste operation will not affect the disabled fields.

### 3.2.2 A newly generated wind turbine model

The figure below illustrates a newly generated wind turbine model. Here we have used the default values in the Turbine Definition dialog box, and selected one of the sample blade designs included with the Fedem WindPower installation.

After generating it, we have selected the turbine in the *Objects* list (i.e., the tree-node labeled “New Turbine” in the Model Manager panel), to display the properties for the generated turbine object in the Property Editor panel. The “New Turbine” node has also been expanded (by first clicking on the “+” sign to the left of the Turbine icon, ) to reveal some of the other underlying components (“Nacelle”, “Gearbox”, etc.) making up the total turbine model.

The Turbine object and many of its sub-components are specializations of macro objects referred to as sub-assemblies. Their role is to create a natural hierarchy of the various model components, to ease navigation within complex models (see [Section 5.15, "Sub-assemblies"](#) for details).



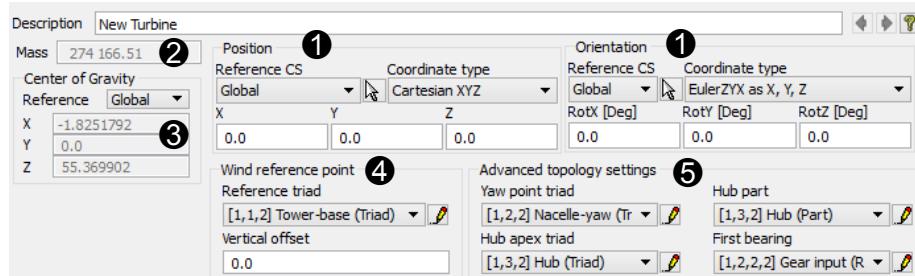
3

In the *Objects* list we see that the turbine model contains a tower, a nacelle, a rotor, as well as joints and triads. Further down the tree, we also see some beam cross sections, materials and so forth. The model tree contains a large hierarchy of different mechanism elements. When selecting any of these elements in the *Objects* list, we see the properties of the selected element in the Property Editor panel. We will describe these elements and properties below. For the basic elements, (parts, triads, joints, etc.), please refer to [Chapter 5, "Mechanism Elements"](#).

### Turbine

The turbine object is, naturally, the main or top-level element in the model. The Turbine includes Tower, Nacelle, Rotor, etc. The Modeler view, in the center of the figure above, shows the various elements of the model in 3D space. We use a scientific visualization to show the 3D position and orientation of each element in the model. Even though you can click the various elements in the modeler, it is probably easier and more logical to access the elements from the tree structure.

The figure below shows the properties of the turbine. They are as follows:



- ① **Position and orientation** - These fields are used to modify the position and orientation of the entire turbine assembly. The X, Y and Z fields specify the position whereas the RotX, RotY and RotZ fields rotate the turbine. Advanced users may wish to explore the Reference CS (reference coordinate system) and Coordinate type fields. See [Section 4.5.4, "Origin property"](#) for a further description of these fields.
- ② **Mass** - This field reports the total mass of the whole turbine assembly. The mass is provided in kilograms when the model is built in SI units. The total mass is updated automatically, whenever the mass property or size of any of the sub-components of the turbine is changed.
- ③ **Center of Gravity** - These fields report the location of the turbine's center of gravity, in either Global coordinates or in Local coordinates with respect to the origin of the turbine assembly. The center of gravity is updated automatically, whenever the mass property or size of any of the sub-components of the turbine is changed.
- ④ **Wind reference point** - These fields specify a Triad and a vertical offset that together define the reference point (origin) for the wind field.
- ⑤ **Advanced topology settings** - These settings are used to correct/reassign certain triads, parts and joints that can become lost when modifying the components of the turbine. If you for example were to delete the tower of the turbine, you would see that the Reference triad and the Yaw point triad will be unassigned. If you later add a tower manually, you would have to assign these fields proper values.

### Tower

The tower of the wind turbine is represented with the Tower object in the *Objects* list. The default tower representation consists of a single Generic Part and a Triad in each end of the tower. A tower model may have to be a bit more complex than that, which is why we offer additional approaches for modeling the tower. Please see [Section 3.4, "Tower definition"](#) for details on these approaches.



**NOTE:** A valid wind turbine model is not required to contain a separate Tower object. The tower may optionally be an integrated part of the underlying support structure. It may then be modeled manually, by importing a FE part or through other means.. In these cases, the turbine model needs to be generated with the tower height set to zero (see bullet ① in [Section 3.2.1, "Turbine definition"](#)).

When the tower object is selected in the *Objects* list, the total mass (in kilograms) of the tower and its center of gravity position are displayed in the property panel, as shown at right. The *Visualize 3D* toggle (active only if the tower consists of beam elements), can be used to quickly toggle on/off the 3D visualization of all the beam elements the tower consists of, see also bullet ③ in [Section 5.3.3, "Beam properties"](#).

Mass	50 000.0
Center of Gravity	
Reference	Global
X	0.0
Y	0.0
Z	24.0
<input type="checkbox"/> Visualize 3D	



**NOTE:** The *Visualize 3D* toggle in the Tower property panel has a tri-state value. In addition to the usual on  and off  states, it can also have a neither-on-nor-off  state, in which the corresponding on or off settings on each element are applied instead. This is usually the default setting for such tri-state toggles.

3

## Nacelle

The nacelle object contains the generator and associated parts, such as the main shaft, secondary shaft, gearbox, etc. Many of the elements in the nacelle will affect the behavior of the wind turbine. Unless there are some limits on the behavior of the turbine, it will for example rotate freely when you apply a wind field. A real turbine offers resistance to a wind field, and has limits on rotational velocity, etc. We will describe the nacelle elements of the turbine below. Fedem WindPower also supports adding a full control system, as we will see in [Section 3.4, "Control system"](#).

## Driveline and Generator

The generator basically contains a revolute joint and its master triad connecting the generator to the nacelle part. The properties of the generator object are illustrated in the figure below.

Mass	0.0	<input checked="" type="radio"/> Torque control	0.0	<input style="width: 20px; height: 20px;" type="button" value="..."/>
Center of Gravity		<input type="radio"/> Velocity control	0.0	<input style="width: 20px; height: 20px;" type="button" value="..."/>
Reference	Global			
X	0.0			
Y	0.0			
Z	0.0			

The generator properties consists of the Mass and Center of Gravity fields, which have similar meaning as for the other turbine sub-components. In addition, it has settings for controlling the torque or

velocity of the generator. The Torque control toggle can be used to assign a load to the generator whereas the Velocity control toggle can be used to set a prescribed velocity. You can enter a specific value in either of these fields. You can also choose a function here which can be created by choosing **Function** on the *Mechanism* menu or the Right-click menu in the *Objects* list (see [Section 5.11.1, "Creating a function"](#)). All existing functions will be shown in the Torque and Velocity control drop lists.

If you expand the tree-list of the generator and select the Generator revolute joint, you will see that the Rz-values of this joint are updated when you change the properties of the generator. You can model more advanced representations or use a control system, as discussed later, if you wish to do more advanced modeling. In this basic introduction, you can leave the properties as they are, add a load, or set the prescribed velocity, if you wish. Adding a load, or using a prescribed velocity, will make the turbine model behave more like a real turbine, since the generator is the main load of a wind turbine.

Depending on the Driveline type selected, the driveline contains:

- *Gearbox* - A driveline with a gear box contains a main shaft and a secondary (high-speed) shaft.
- *Direct* - A direct driveline contains a main shaft only.

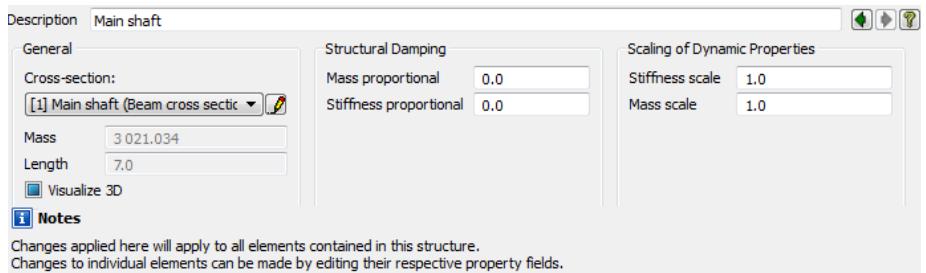
The driveline may optionally have zero, one or two bearings, depending on what you selected when you created the turbine. The main shaft will contain one, two or three beam elements depending on the number of bearings added.

The gear box driveline also has a gearbox object that contains:

- *A gear* - The gear basically specifies the transmission output ratio. You will find the gear inside the Gearbox in the tree-list. When you select the gear, the properties will display the transmission output ratio.
- *Two revolute joints* - The gearbox includes two revolute joints, one at each end of the gear.
- *Two triads* - The gear box also includes two master triads for each joint, connecting the gear box to the nacelle part.

## Shaft

The figure below illustrates the properties for the main and secondary shafts. It is essentially identical to the property panel of the Beam elements, described in [Section 5.3.3, "Beam properties"](#), except for the Notes in the bottom of the frame, and that the *Visualize 3D* toggle here has a similar tri-state behavior as explained above for the [Tower](#).



3

The *Mass* and *Length* fields of the *Shaft* property panel display the sum of the corresponding fields of all the beams that the shaft contains.

Editing any of the *Structural Damping* or *Scaling of Dynamic Properties* fields, the values entered will apply to all the underlying beam elements of the shaft, which can be verified by selecting any of the beam elements. But, if such a field is edited for one beam element such that the beams making up a shaft no longer have identical properties, the corresponding field in the *Shaft* property panel will be blank (no value displayed).

There are also additional elements in the nacelle object, such as the nacelle *Part* representing the nacelle itself, the revolute joints for the bearings, and various triads. See [Chapter 5, "Mechanism Elements"](#) for a full description of those object types.

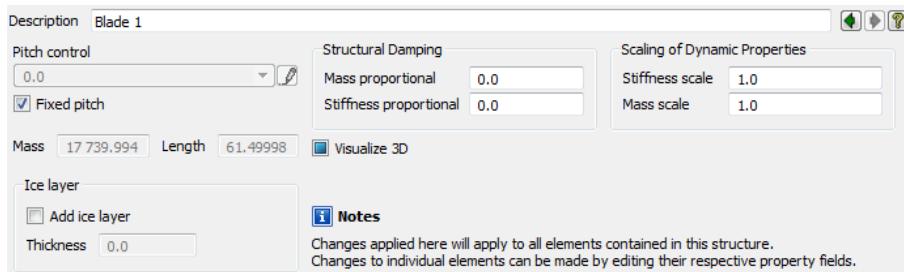
## Rotor

The rotor basically consists of the blades and the hub. Fedem WindPower enables you to design your own blades, using any number of Airfoil data sets for the blades, to set the air environment, etc., as we will see in [Section 3.4, "Creating an advanced wind turbine model"](#). But, in this basic introduction, we just use a typical three-blade configuration using the sample blade design "Sample\_5MW" that comes with the installation.

In the *Objects* list, we see that the rotor contains the following:

- **Three blades** - These objects represent the turbine blades. Beneath each blade you will find a list of beams and triads that the blade consists of. We will look closer at the blade modeling in [Section 3.4.2, "Blade definition"](#).
- **Three revolute joints** - These are the pitch joints of the blades. The pitch controller basically adjusts the angular motion in these joints.
- **A rigid joint** - This is the joint that connects the main shaft to the hub part in the rotor.
- **The hub part** - This is a Generic part representing the hub itself. Each arm of the hub is connected to a pitch joint.
- **Four triads**- These are triads used to connect the parts and joints.

The figure below illustrates the properties for a blade object.



The *Pitch control* is used to control the pitching angle. It can either be fixed or prescribed. In the latter case, either a constant value or a function can be assigned to describe the actual pitch angle. The setting of the *Pitch control* field is also reflected in the Rz-values of the corresponding revolute joint, which can be edited if more advanced pitch modeling is needed, e.g., adding flexibility and damping in the pitching motors, etc.

The blade property panel also include fields for total mass and length, which are useful for verifying the model before simulation. It also has a *Visualize 3D* toggle with similar operation as for the *Tower* and *Shaft* objects described above, as well as *Structural Damping* and *Scaling of Dynamic Properties* fields with similar functionality as for the *Shaft*.

You can also add a ice layer, resulting in extra non-structural mass on the blade. The ice mass is estimated based on a mass density of  $0.917 \text{ kg/m}^3$  and an ellipsoidal cross section, using the specified *Chord length* and *Thickness ratio* parameters of the [Blade definition](#) to deduce the length of the principal axes of the ellipse.

### 3.3 Basic solving and analysis

After creating a basic wind turbine model, we can “solve” that model.

If you are new to this, you are probably wondering what “solving” is. Well, solving is basically the process of (1) mathematically reducing the turbine model into a smaller and easier to manage internal model that can be solved numerically, and then to (2) make an animated physics model of the wind turbine.

After running the Fedem dynamics solver, you will be able to play an animation where you can see the effect of all loads and environmental conditions acting on the turbine. If you, for example, have used a too thin tower, you may see it break down when the wind speeds get too high.

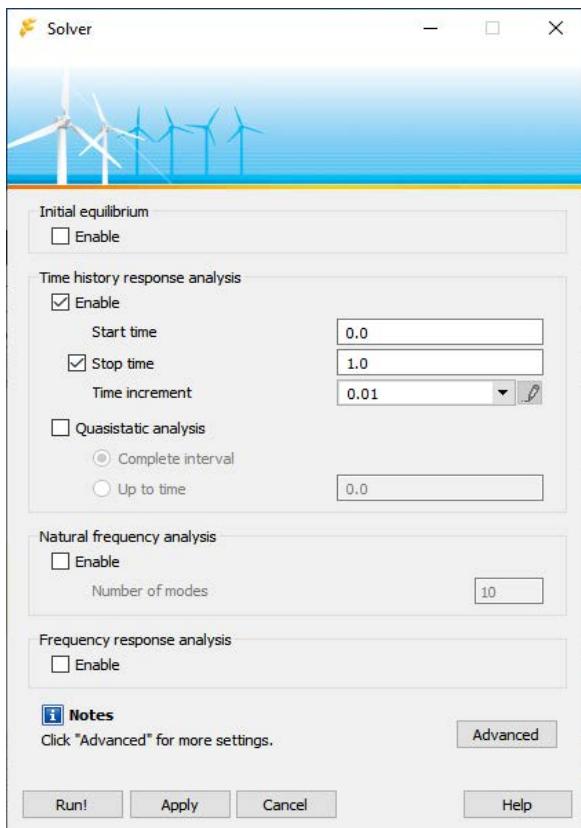
Fedem WindPower also contains tools like Simulation events that enables you to set up and test a range of changing conditions, to see how the wind turbine behaves in response to these events.

We will only describe basic solving in this section, whereas [Section 8.5, “Dynamics analysis”](#) contains full details about the solving process. It is recommended to read that part before using the results in an actual project. Good and usable results are highly dependent on the analysts experience, and correct application of modeling, solving and analysis.

### 3.3.1 Running the dynamics solver (basic mode)



You can display the Dynamics Solver Setup dialog box (shown below) by choosing **Dynamics Solver (Basic Mode)...** on the *Solve* menu, or by clicking the associated button on the *Solvers* tool bar.

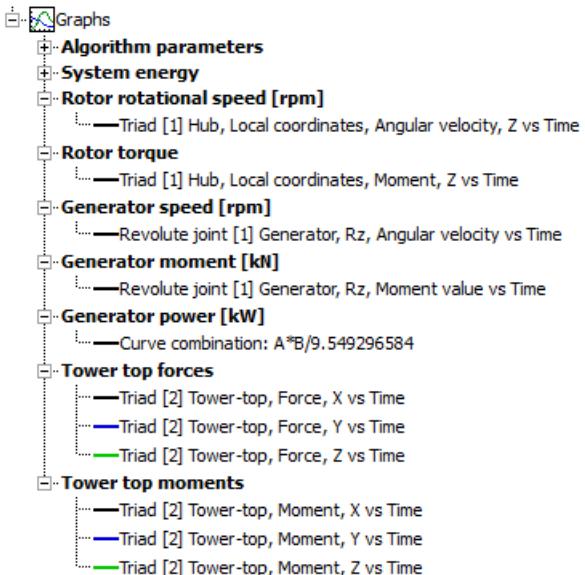


In this dialog box you can adjust the *Start* and *Stop time* of the simulation, as well as the *Time increment* size to use. Please refer to [Section 8.5.1, "Dynamics Solver \(Basic Mode\)"](#) for details on all fields of the dialog box. Wind turbine simulations usually need to be performed over a relatively long time span and the default stop time is typically 1.0. Therefore, at least you will need to edit this field before starting the solver.

When finished defining the simulation setup, click the **Run!** button to start the dynamics solver. The solver will then run in the background as a separate process, such that you can continue browsing the model, etc., while it is running. It is not possible to modify anything though, when solver has started producing results data on secondary storage.

### 3.3.2 Basic results overview

If everything went well, and no error messages have been reported in the output list, then we will now have a result database that can be looked at in more detail. If we click the *Results* tab on top of the Model Manager panel on the left side of the main window, we will see a set of predefined curves under the *Graphs* node, as shown below.

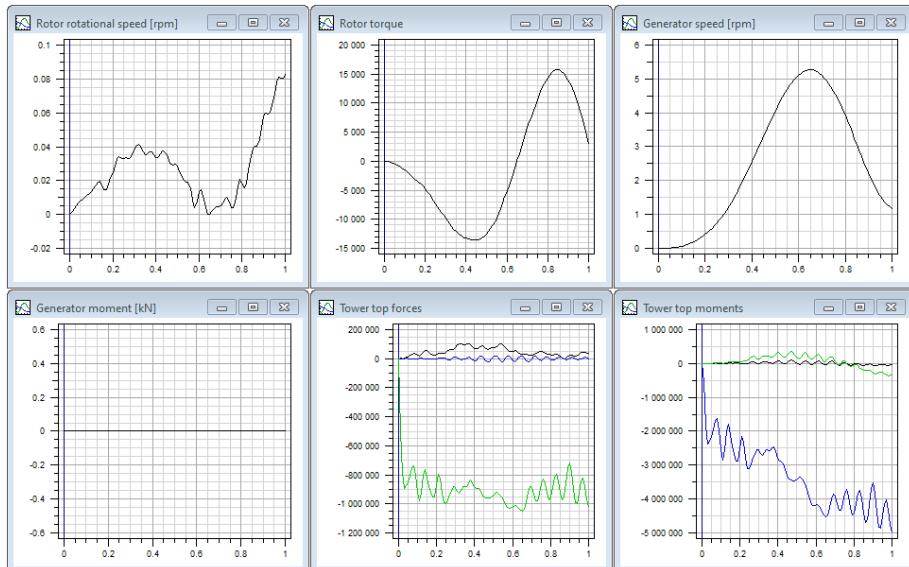


These curves are as follows:

- *Algorithm parameters* - Contains curves monitoring the solution algorithm performance (not important for our basic results review).
- *System energy* - Contains curves plotting various system energy quantities, to verify global energy conservation (not important now).
- *Rotor rotational speed [rpm]* - The angular velocity of the turbine rotor.
- *Rotor torque* - The torque of the rotor.
- *Generator speed [rpm]* - The rotational speed of the rotor.
- *Generator moment [kN]* - The generator moment.
- *Generator power [kW]* - The generator power generation.
- *Tower top forces* - Force components on the top of the tower.
- *Tower top moments* - Moment components on the top of the tower.

You can right click a graph (bold text) or a curve under a graph to choose *Show Graph*. If no graph is displayed, it might be due to a solver error. In that case, refer to [Section 8.5, "Dynamics analysis"](#) for trouble-shooting.

The figures below illustrate the graphs that will be displayed when you choose to show them after running the dynamics solver. To learn more on graphs and curves in Fedem, see [Section 9.2, "Graphs"](#).



### 3.3.3 Reliability and validity of results

The reliability and validity of the results is a very important topic. It is the highest priority for Fedem to ensure reliable and valid results. *Reliability* is the consistency of a result, or the degree to which a result provides the same results every time it is used under the same conditions. *Validity* is focused on whether a result is valid.

Fedem WindPower uses AeroDyn for calculation of the forces due to wind on the blade elements. AeroDyn bases its calculations on provided input on the air environment, updated positions and velocities of the wind turbine, and airfoil tables with drag and lift coefficients. The output forces on the blade elements are used by our solvers to provide results that can be analysed. The reliability and validity evaluations/certifications of the AeroDyn implementation are available at the National Renewable Energy Laboratory (<http://wind.nrel.gov>).

## 3.4 Creating an advanced wind turbine model

One of Fedem WindPower's greatest strengths is the ability to do very advanced and highly customized modeling tasks, while at the same time being easy and effective to use. We will in this section look in more detail at some advanced wind turbine modeling issues which may be used to customize the wind turbine model before the numerical simulation is performed. These are as follows:

- *Airfoil definition* - The Airfoil Definitions dialog box specifies the aerodynamic cross section properties of the wind turbine blades. The airfoil data is stored on external files and used by the *Blade Definitions* to describe the design of the turbine blade to be used. Since these files are read directly by AeroDyn their format is accordingly.
- *Blade definition* - The Blade Definition dialog box specifies a wind turbine blade design. Both aerodynamic and structural properties are specified here. The aerodynamic properties are used by AeroDyn to calculate forces on the turbine blades which are applied as loads on the structural model.
- *Aerodynamic setup* - The Aerodynamic Setup dialog box is used to specify the global air environment for the wind turbine, that is, the specification of the wind field to use, deterministic or turbulent. It also contains some fields with control parameters for AeroDyn.
- *Sea environment* - The Sea Environment dialog box is used to specify the sea environment for an offshore wind turbine. We can here specify the mean sea level, water depth, water density, functions for waves and current, etc. Refer to [Section 6.1, "Sea environment"](#) for a detailed description of this dialog box.
- *Tower definition* - The Tower Definition dialog box is used to define a more advanced tower model than the default tower representation generated via the *Turbine definition* dialog box. The Tower Definition dialog box offers an intermediate complex turbine tower. Even more advanced towers can be modeled by importing FE-models.
- *Control system* - The Control system is used to make functions that can monitor and modify properties in the wind turbine model. The wind turbine that is created by the *Turbine definition* dialog box can use an optional default control system, that makes the turbine behave more like a real wind turbine with limited rotational speed, pitch controller, torque controller, etc.

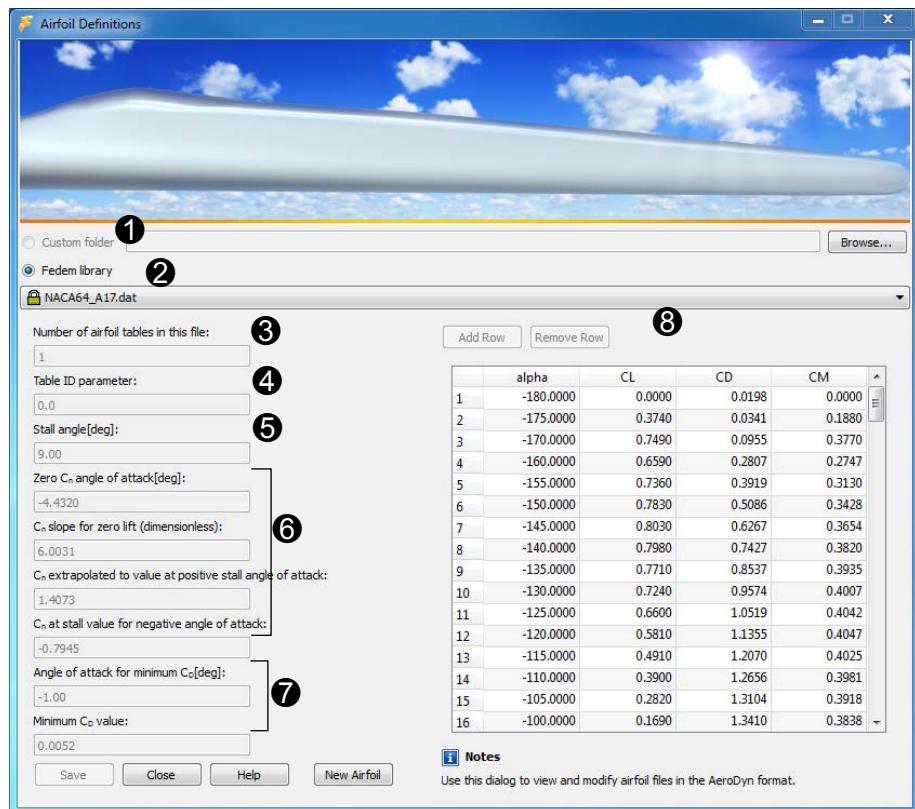
### 3.4.1 Airfoil definition

The purpose of the Airfoil Definitions dialog box is to facilitate the creation of a set of airfoils to use in the turbine blade definition. An airfoil is basically a cross section of the blade where different aerodynamic properties (drag and lift coefficients, etc.) are specified around the blade.

Fedem WindPower uses the AeroDyn airfoil format of NREL. The Airfoil Definition dialog box is simply an editor for such files. For information on this format, see <http://wind.nrel.gov/designcodes/simulators/aerodyn>.

The airfoil files are referred to when designing the blades, in the *Blade definition* dialog box, and when the completed turbine model is being solved, the used airfoil files will be read by AeroDyn (via the dynamics solver) to calculate the wind loads.

The Airfoil Definition dialog box is shown below:



- ❶ These two radio buttons allow you to select whether to use the airfoils provided with the installation (*Fedem library*), or from a *Custom folder*, that may be selected using the **Browse...** button.
- ❷ This drop-down menu allows you to select the name of the airfoil file to be used. If using airfoils from the *Fedem library*, their properties can not be edited, which is indicated by the lock symbol.
- ❸ *Number of airfoil tables in this file* - The airfoil file format is designed so that they can contain several actual airfoil tables. The airfoil file in the figure above contains a single table only.
- ❹ *Table ID parameter* - Identifier for each airfoil table.
- ❺ *Stall angle [deg]* - The stall angle, in degrees, of this airfoil.
- ❻ The next four parameters characterize the normal force coefficient ( $C_n$ ). Refer to the User's Guide and the AeroDyn Theory Manual of AeroDyn for the interpretation of these parameter values.
- ❼ The next two parameters specify the minimum drag coefficient ( $C_D$ ). These two values can be deduced from the tabulated  $C_D$ -values on the right side of the dialog box.
- ❽ On the right side of the dialog box, we find the airfoil table data with the following columns:
  - $\alpha$  - The angle of attack. The table must be written in order of increasing angle of attack. It is preferable to use a table for angles between -180 to +180 degrees. The coefficient values should be the same at -180 and +180 degrees to avoid discontinuity.
  - $C_L$  - Static lift coefficient for this angle of attack.
  - $C_D$  - Drag coefficient for this angle of attack.
  - $C_M$  - Pitching moment coefficient for this angle of attack.

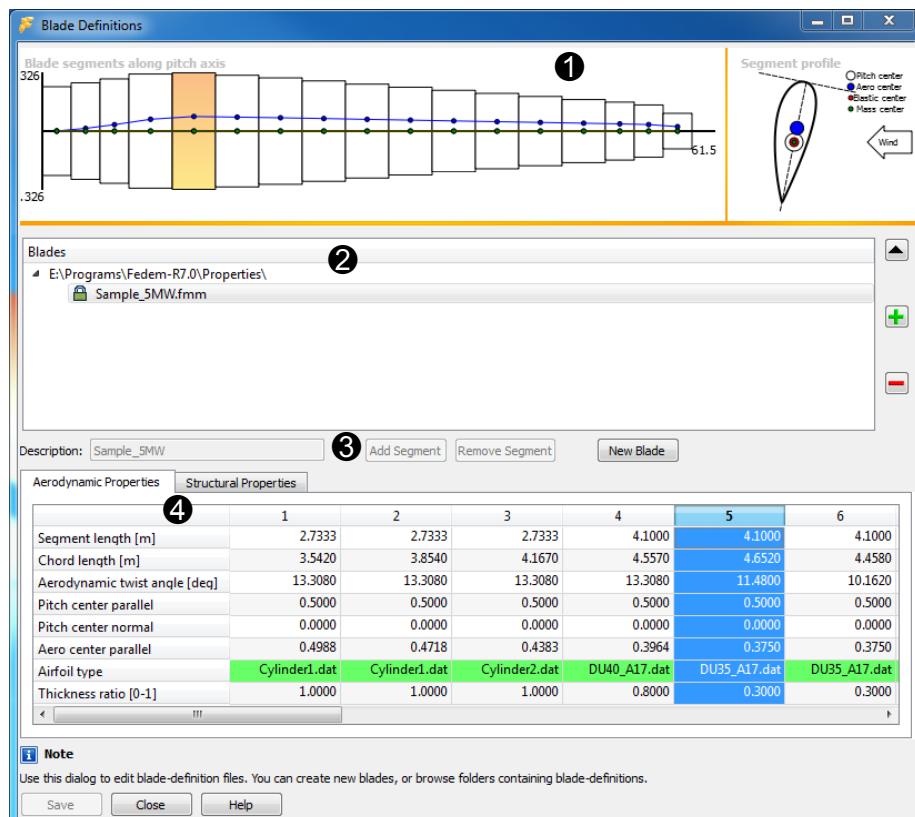
The **Add Row** button adds an empty row (all values zero) at the bottom of the table. The **Remove Row** button deletes the last row.

The **New Airfoil** button creates a new, empty, airfoil and is placed in the current *Custom folder* (you can't create new airfoils in the *Fedem library*). The **Save** button writes the current dialog box content to the airfoil file. For more information, please see the AeroDyn User's Guide, available from the web site <http://wind.nrel.gov/designcodes/simulators/aerodyn>.

We have included a set of sample airfoil files, provided by NREL, with the Fedem WindPower installation. In some projects, the actual blade design is of less concern, and then using these airfoils will be satisfactory. But, in other projects, the user may wish to provide their own airfoils.

### 3.4.2 Blade definition

Fedem Windpower includes some sample blade designs (e.g., the file "Sample-5MW") within the installation. You can just specify one of these sample blades in the *Turbine definition* dialog box, if the blades of the turbine are not the main focus area for your simulation. However, most projects will require that the blades are defined in detail for that project. This is performed in the Blade Definition dialog box, shown below.



The Blade Definition dialog box consists of four main parts:

- ① *Blade visualization area* - The blade segment length and Chord length are visualized in the top left side of the dialog box, and the blade cross section for the selected segment is visualized in top right side.
- ② *Blades area* - Beneath the blade visualization, a list of file paths to the available blade designs are shown. Just click on a file name to display its properties in the visualization area and in the data table below.

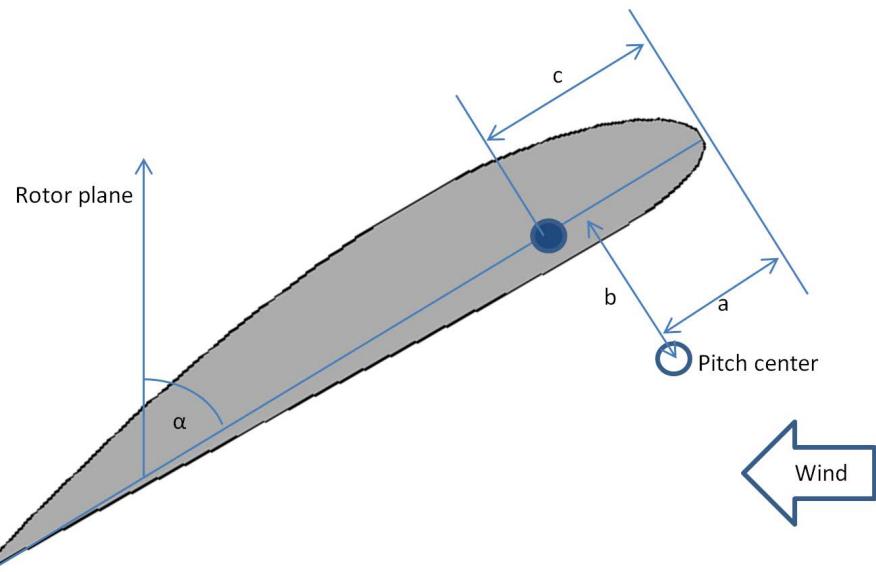
Use the green + button and the red - button to the far right to browse for new libraries of blade designs, or to remove a selected folder from the *Blades* list. Blades that are included with the installation are not editable, as indicated by the lock symbol.

- ③ **Description** - In the description field you can assign a name to the blade. Next to it you find the **Add Segment** and **Remove Segment** buttons, which are used to add a new column in the data table, and to remove the currently selected column (or the last one if none were selected), respectively. The **New Blade** button can be used to create a completely new, initially empty, blade design.
- ④ **Data table** - The table in the lower part of the dialog box contains the *Aerodynamic* and *Structural Properties* in separate tabs (see below). The blades are discretized into a limited number of segments with constant cross section properties, and each column of the table represents one blade segment. If you click on a column, its properties are illustrated in the upper right part of the dialog box.

The figure above shows the Blade Definition dialog box when it displays the *Aerodynamic Properties* of the blade segments. It consists of the following rows:

- *Segment length [m]* - The length of each blade segment, along the pitch axis of the blade.
- *Chord length [m]* - Width of the blade segment (perpendicular to the length of the blade), edge to edge.
- *Aerodynamic twist angle [deg]* - Twist (in degrees) of the blade for this segment (labeled  $\alpha$  in the figure below), relative to the rotor plane.
- *Pitch center parallel* - Location of the pitch center along the chord line (labeled  $a$  in the figure below), given as a fraction of the chord length. This value is typically in the range [0,1], but can be negative if the center is located in front of the leading edge, and larger than one if it is located behind the trailing edge.
- *Pitch center normal* - Pitch center offset from the chord line in the normal direction ( $b$ ), given as a fraction of the chord length.
- *Aero center parallel* - Location of the aero center along the chord line ( $c$ ) relative to the leading edge, given as a fraction of the chord length.
- *Airfoil type* - File name of the airfoil data for this blade segment. If you click on the file name a browse button [...] appears besides it, and by clicking that one you can browse for another airfoil file. See [Section 3.4.1, "Airfoil definition"](#) above for more details on the airfoils.

- **Thickness ratio [0-1]** - Thickness of the blade relative to the chord length. This value is only used to estimate the circumference of the cross section when calculating the weight of ice on the turbine blades (see "Rotor" in Section 3.2.2). It does not affect the aerodynamic load calculations. In addition, this value is also used to create the cross section visualization in the upper right part of the dialog box.

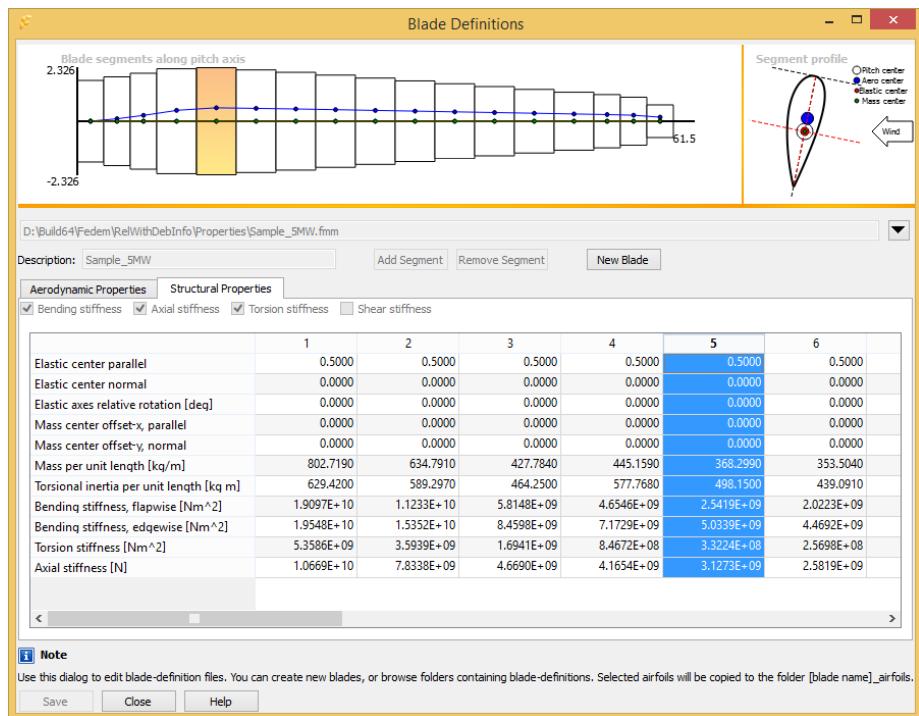


The figure below shows the Blade Definition dialog box when it displays the *Structural Properties* of the blade segments. In this figure, the *Blades* area has also been contracted to show only the path of the currently selected blade design. It can be expanded again (as in the figure on page 3-22) by clicking the black triangle to the right of the file path.

Above the *Structural Properties* table, there are four toggles that can enable or disable the different stiffness terms. When disabled, the associated rows are hidden from the table.



**NOTE:** The purpose of the stiffness toggles is to provide a simple means to create a structural model of the blade when not all stiffness properties are known. By disabling either the Bending, Axial or Torsional stiffness fields, their actual values are replaced by some internally computed large values, large enough to ensure that the associated deformations become negligible. For the shear stiffness toggle, this is not needed, as the beam formulation used allows for neglection of shear deformation simply by setting the shear stiffnesses to zero.



When all the stiffness toggles are enabled, the *Structural Properties* table consists of the following rows:

- *Elastic center parallel* - Location of the elastic center (also known as the neutral axis) along the chord line, given as a fraction of the chord length. This value is typically in the range [0,1], but can be negative if the center is located in front of the leading edge, and larger than one if it is located behind the trailing edge.
- *Elastic center normal* - Elastic center offset from the chord line in the normal direction, given as a fraction of the chord length.
- *Elastic axes relative rotation [deg]* - Angle (in degrees) between the chord line and the local y-axis of the structural beam element, for which the *flap-wise* bending stiffness is referring to. The *edgewise* bending stiffness is then referring to the local z-axis, defined via the cross product of the length axis of the beam and the local y-axis.
- *Mass center offset, parallel and normal* - These two values specify the offset of the mass center from the elastic center, along the rotated elastic axes, given as fractions of the chord length.

- *Mass per unit length [kg/m]* - Mass density of this blade segment.
- *Torsional inertia per unit length [kg m]* - Torsional inertia of this blade segment.
- *Bending stiffness, flap-wise and edge-wise [Nm<sup>2</sup>]* - These two values specify the bending stiffnesses  $EI_{yy}$  and  $EI_{zz}$ , respectively, with respect to the local elastic axes defined by the *Elastic axes relative rotation*.
- *Torsion stiffness [Nm<sup>2</sup>]* - The torsional stiffness.,  $Gl_t$ .
- *Axial stiffness [N]* - The axial stiffness,  $EA$ .
- *Shear stiffness, flap-wise and edge-wise [N]* - These two values specify the shear stiffnesses  $GA_{sy}$  and  $GA_{sz}$ , respectively, with respect to the local elastic axes.
- *Shear center, parallel and normal* - These two values specify the offset of the shear center from the elastic center, along the rotated elastic axes, given as fractions of the chord length.

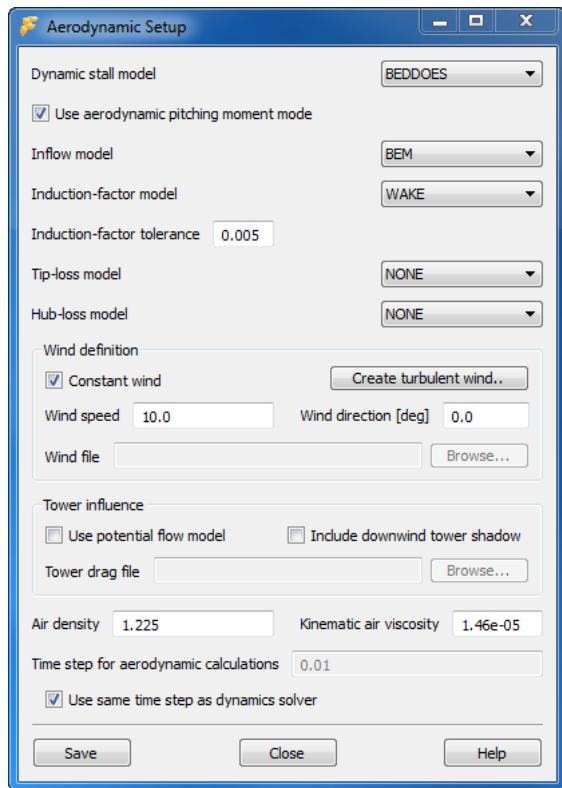
The stiffness parameters described above are the same as specified for *Generic beam cross sections*, see [Section 5.4, "Beam cross sections"](#).

When a turbine model is generated using the **Generate Turbine** button of the [Turbine definition](#) dialog box, two Beam elements (with common cross section properties) and three Triads are created for each blade segment. The middle triad becomes the target node of the aerodynamic forces computed by AeroDyn on that segment, and the other two triads are only the connection points to the neighboring blade segments.

The aerodynamic forces are computed at the aero center and are then transformed onto the middle Triad on the pitch axis, using the specified offset parameters ( $a, b, c$  in the figure on page 3-22). The velocity and position of the aero centers are then updated based on the state variables of the Triad, using the inverse transformation.

### 3.4.3 Aerodynamic setup

The Aerodynamic Setup dialog box (shown below) is used to define the air environment for the wind turbine, and to set up some control parameters for the aerodynamic load calculation in AeroDyn. An input file to AeroDyn is generated based on the contents of this dialog box.



3

- **Dynamic stall model** - Specify BEDDOES to use the Beddoes-Leishman dynamic stall model, or STEADY for quasi-steady airfoil characteristics. It is recommended to use the BEDDOES model in most situations.
- **Use aerodynamic pitching moment mode** - This option enables the calculation of aerodynamic pitching moment. The Airfoil files must then include pitching moment coefficients (CM).
- **Inflow model** - This option controls the dynamic inflow model. Specify GDW to use a Generalized Dynamic Wake inflow model to calculate the induction factor, or BEM to use the Blade Element Momentum theory with skewed wake and tip loss corrections. The direct calculation method of the GDW option is considerably faster than the

iterative method of the BEM option, which assumes that the wake is always in equilibrium with the forces on a blade element. See Appendix E of the AeroDyn User's Guide for more information on the dynamic inflow method employed with the GDW option.

- *Induction-factor model* - This option controls the wake or induced velocity calculation. There are three possible choices, WAKE, SWIRL and NONE. Specify SWIRL to calculate both axial and tangential induction. If WAKE is specified, only the axial induction will be calculated. If NONE is specified, the induced velocity calculation will be completely bypassed and all induction factors reset to zero. This option is available primarily to assist the debugging of new models. We suggest that in the first tests of a new model, ignore the wake to accelerate the calculations and eliminate the possibility of convergence problems in the induction factor iteration. A warning is generated by AeroDyn when NONE is used to remind the user that this is a highly unusual situation.
- *Induction-factor tolerance* - This is the tolerance used for convergence testing in the iterative solution to find the induction factor,  $\alpha$ , when using the GDW inflow model. The default value 0.005 should be used unless there are compelling reasons to do otherwise. Sometimes it is desirable to change this value to avoid convergence problems (with some loss of accuracy) or to speed up the calculations. The value represents the maximum allowed difference between two successive estimates of  $\alpha$ . That is, if the new estimate of  $\alpha$  differs from the estimate from the previous iteration by an amount less than the tolerance, the solution has converged and the last value of  $\alpha$  is used.
- *Tip-loss model* - This option controls the tip loss model (used with the BEM inflow model only). There are three possible choices, PRAND, GTECH and NONE. PRAND is the Prandtl tip loss model, GTECH is the Georgia Tech correction to the Prandtl model, whereas NONE turns off the tip loss correction altogether. The GTECH model is intended to better model the effects of the relatively large inflow velocities (compared to hovering rotors, for which the Prandtl model was developed) experienced by wind turbine rotors spacing the tip vortex rings farther apart in the wake.
- *Hub-loss model* - This option controls the hub loss model (used with the BEM inflow model only). There are two possible choices, PRAND, and NONE. PRAND invokes the Prandtl tip loss model, to be used to determine hub losses, whereas NONE turns off the hub loss correction altogether. This option is intended to model losses experienced by the rotor blade elements close to the rotor hub. Generally, these effects are of little consequence.

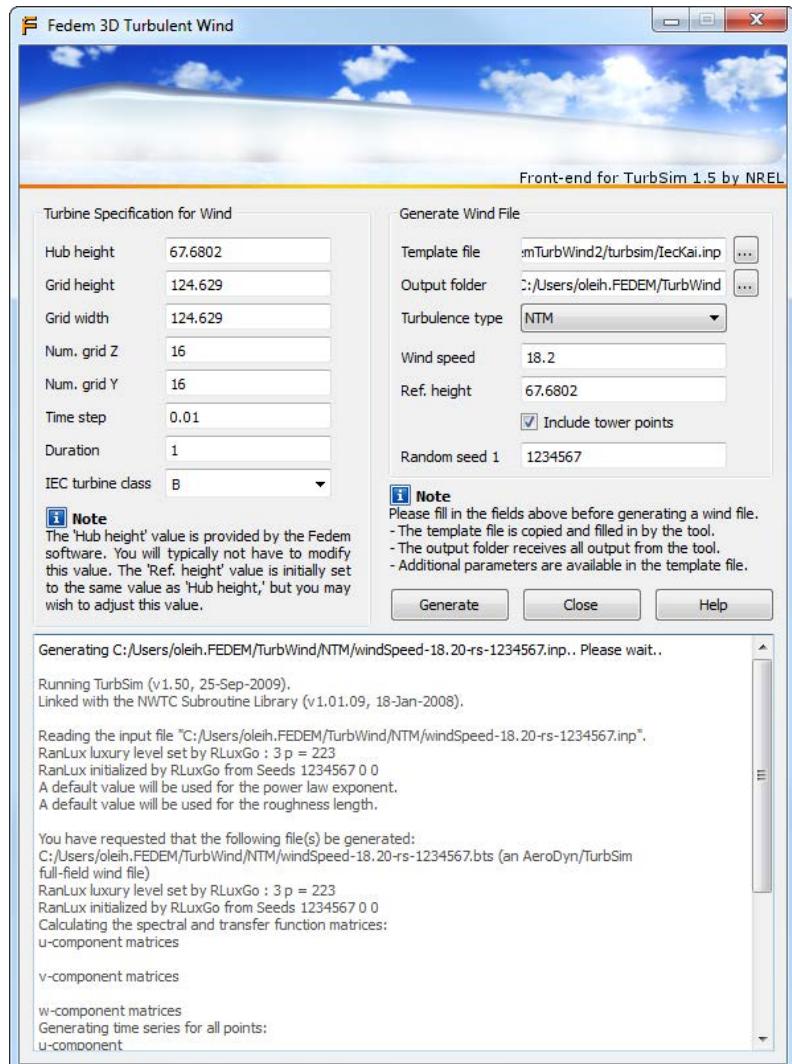
- **Wind definition** - This part of the dialog box is used to define the undisturbed wind field to be used in the aerodynamic load calculation.
  - *Constant wind* - Enable this toggle to specify a constant wind using the *Wind speed* and *Wind direction* fields. The wind direction value is the angle (in degrees) between the rotor plane normal (with zero shaft tilt) and the wind direction, see Section 7.1 in the AeroDyn User's Guide.
  - *Wind file* - When the *Constant wind* toggle is disabled, you may use the **Browse...** button to select a separate input file describing the wind field. This can either be hub-height wind data file, or a full-field turbulent wind data file, described in Chapter 7 and 8, respectively, in the AeroDyn User's Guide.
  - *Create turbulent wind..* - Click this button to launch the Fedem TurbWind tool to create a full-field turbulent wind data file for the current model, see "[Turbulent Wind Definition](#)" below. After running the tool, use the **Browse...** button to select the resulting turbulent wind file.
- **Tower influence** - This part of the dialog box is used to set up the tower influence on the wind field.
  - *Use potential flow model* - Enables the use of a tower influence model based on potential flow solution around a cylinder, as described in the AeroDyn Theory Manual. This model provides the influence of the tower on the local velocity field at all points around the tower, including increases in wind speed around the sides and the cross-stream velocity component in the tower-near flow field.
  - *Include downwind tower shadow* - Enables the use of a tower wake (velocity deficit) model for downwind rotors. This toggle has no effect for upwind turbines.
  - *Tower drag file* - When one or both of the two tower influence toggles are enabled, a tower drag file must be specified. It lists the drag coefficient for the tower as function of the Reynold's number and the tower diameter.
- **Air density** - Specifies the ambient air density.
- **Kinematic air viscosity** - Specifies the kinematic air viscosity. This value is used in AeroDyn to calculate the local element Reynold's number, which can be used to move between multiple tables in the airfoil files.

- *Time step for aerodynamic calculations* - Specifies the time step size ( $dtAD$ ) used by AeroDyn. Since the time integration steps in the dynamics solver ( $dt$ ) often are quite small, the dynamics simulation can run faster and be more immune to numerical stability problems if  $dtAD$  is specified larger than  $dt$ , but still less than the time scale for the changes in aerodynamic forces. Typically, the aerodynamic forces should not be expected to change faster than the time it takes the blade to rotate 2.4°, e.g., if a rotor runs at 30 rpm, it will take 0.02 seconds for the blade to move 3.6° such that  $dtAD = 0.02$  will then repeat the aerodynamic calculations often enough to catch the true variations in the loads. Since the time stepping is controlled by the Fedem dynamics solver, the aerodynamic calculations will be repeated after a time interval that is at least  $dtAD$  (but less than  $dtAD + dt$ ). That is, the aerodynamic calculations are not necessarily repeated exactly at the intervals  $dtAD$ .
- *Use same time step as dynamics solver* - Enable this toggle to ignore the *Time step for aerodynamic calculations* value and always use the same step size in AeroDyn as in the dynamics solver, i.e.,  $dtAD = dt$ .

### Turbulent Wind Definition

The Turbulent Wind Definition dialog box is used to generate turbulent wind files for Fedem WindPower. It is basically a front-end for the TurbSim tool by NREL (<http://wind.nrel.gov>). Fedem WindPower sends the parameters of the wind turbine to TurbSim, such as hub height, grid height, time step, duration, output folder, ref. height, etc. You should adjust these fields to match your wind turbine and conditions.

The figure below illustrates the Turbulent Wind Definition dialog box:



- **Hub height [m]**- Specifies the hub height of the turbine for which the inflow is being generated. This parameter is used as a reference height for determining the grid location.
- **Grid height [m]**- This parameter is the distance between the top and bottom of the grid. The top of the grid is assumed to be aligned with the top of the rotor disk (see TurbSim User's Guide), and because all points of the grid must be above ground level,  
 $0.5 * \text{GridHeight} < \text{HubHeight}$ .

- *Grid width [m]*- This parameter is the width of the grid. The rotor is assumed to be centered horizontally on the grid. If you are generating FF files for AeroDyn, the grid width—like the height—must be large enough to ensure that no part of the blade lies outside the grid, even when the system is displaced.
- *Num. grid Z*- Specifies the number of grid points to generate in the vertical direction. It must be an integer greater than 1.
- *Num. grid Y*- Specifies the number of grid points to generate in the horizontal direction. It must be an integer greater than 1.
- *Time step [s]*- The TimeStep parameter is the time step. It is set to 0.05 seconds in the sample input files, and that value is recommended for most simulations. The time step determines the maximum frequency,  $f_{max}=1/\text{TimeStep}$ , used in the inverse FFT.
- *Duration [s]*- Duration sets the analysis time and usable time in the TurbSim input file (see TurbSim User's Guide).
- *IECturbine class*- IEC turbulence characteristic ("A", "B", "C" or TI in %) or KHTEST
- *Template file*- TurbSim inp-file to use as template.
- *Output folder*- The folder where output files will be stored.
- *Turbulence type*- This parameter indicates which IEC wind model will be used. Valid entries, which are found in the TurbSim User's Guide, include the Normal Turbulence Model (NTM), Extreme Turbulence Model (ETM), and Extreme Wind Speed Model (EWM) using the 10-minute average wind speed with a recurrence period of 1 year or 50 years. Note that the EWM scaling parameters in TurbSim are valid only for 10-minute simulations. The definitions of these models and of the wind turbine classes can be found in the IEC 61400-1 standard (3rd ed.). If the IEC turbine class (IECturbc) parameter was specified as a percentage instead of as a standard turbulence category, the wind model must be "NTM". This input is used only with the IEC spectral models.
- *Wind speed*- Specifies the mean stream-wise wind speed at the reference height. It is the mean value over the entire analysis time length of the simulation of the u-component wind speed. It must be a positive value in units of meters per second.
- *Ref. height [m]*- Specifies the height of the corresponding reference wind speed. This parameter enables users to specify the mean wind speed at a height other than the hub height. TurbSim uses this reference height and wind speed with the wind profile type to calculate the HH mean wind speed.

- *Include tower points*- Determines whether TurbSim generates binary tower time series, which contain points in a line at the tower centerline from the bottom of the rectangular grid to the ground.
- *Random seed 1*- This input parameter is used in conjunction with the next parameter, RandSeed2 in the inp-file; it tells TurbSim how to initialize the pRNG. This random seed must be an integer between -2147483648 and 2147483647 (inclusive).  
The random numbers generated by the pRNG are used to create random phases (one per frequency per grid point per wind component) for the velocity time series. When the pRNG is initialized in the same way (i.e., RandSeed1 and RandSeed2 are not changed), the user can reproduce the same random phases between runs, which is useful in comparing the effects of changes to other input parameters. Random numbers also are used to generate some default input values and the superimposed coherent structures for the non-IEC spectral models.

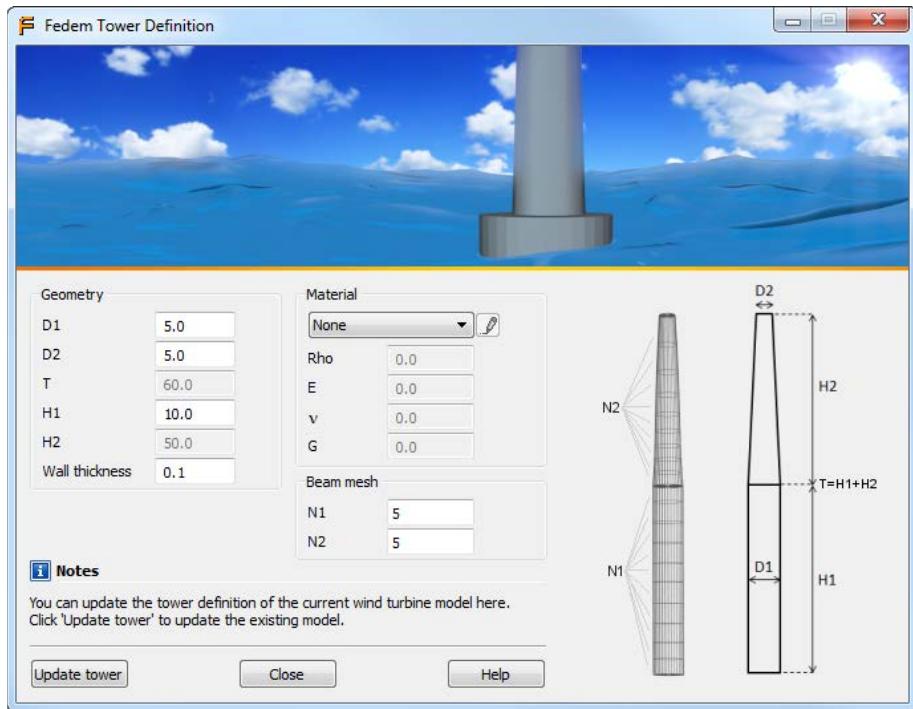
Click the **Generate** button to run TurbSim for generating output files in the output folder. The console output from TurbSim is listed in the lower text area of the dialog box.

#### 3.4.4 Tower definition

The Tower Definition dialog box is used to create a more complex tower based on the following fields:

- *D1 [m]* - Specifies the diameter of the tower at the top.
- *D2 [m]* - Specifies the diameter of the tower at the bottom.
- *T [m]* - Displays the total tower height.
- *H1 [m]* - Specifies the height of the lower part of the tower.
- *H2 [m]* - Displays the height of the upper part of the tower.
- *Wall thickness [m]* - Specifies the thickness of the wall.
- *Material* - Specifies the tower material.
- *Rho, E, ν, G* - Display the properties of the specified tower material.
- *N1* - Specifies the number of beam elements in the lower half.
- *N2* - Specifies the number of beam elements in the upper half.

The figure below illustrates the Tower Definition dialog box.



The Tower Definition dialog box is available only if the initial turbine model was generated with a tower height larger than zero. Otherwise, the tower (if needed), has to be added manually, by importing a separate FE part (see [Section 5.1.2, "Creating parts by file import"](#)) or a spaceframe (see [Section 6.5.1, "Import of space frames"](#)), and manually attaching it to the master triad of the yaw joint (see "[Attaching joints](#)" in [Section 4.6.1](#)).

### 3.4.5 Control system

Fedem WindPower has support for advanced control system modeling. See [Chapter 7, "Control System Modeling"](#) for more details.

The [Turbine definition](#) dialog box includes a check box for including a control system with the wind turbine. This control system provides:

- A low-pass filter on the generator rotation velocity.
- A pitch controller for the z-rotation.
- A general pitch controller.

The figures below show the properties of the control system functions:

**3**

The image shows three separate configuration dialog boxes, each consisting of a left panel for 'Function Type' and a right panel for 'Parameters'.

- Low-pass filter:**
  - Function Type:** Low-pass filter
  - Argument x:** [1] Generator (Revolute joint) (selected), DOF: Z rot., Var: Velocity
  - Argument y:** Time, DOF: , Var:
- Pitch controller:**
  - Function Type:** Pitch controller
  - Argument x:** [1] Pitch 1 (Revolute joint) (selected), DOF: Z rot., Var: Length/angle
  - Argument y:** [1] Filtered velocity (Function), DOF: , Var:
  - Argument z:** Time, DOF: , Var:

Parameter	Value
Reference speed [rad/s]	122.9096
Integral gain, Ki	0.008068634
Pitch for doubled power, Kk [rad]	0.1099965
Proportional gain, Kp [s]	0.01882681
Minimum pitch setting [rad]	0.0
Maximum pitch setting [rad]	1.570796
Maximum pitch rate [rad/s]	0.1396263
- Torque controller:**
  - Function Type:** Torque controller
  - Argument x:** [2] Pitch controller (Function), DOF: , Var:
  - Argument y:** [1] Filtered velocity (Function), DOF: , Var:
  - Argument z:** Time, DOF: , Var:

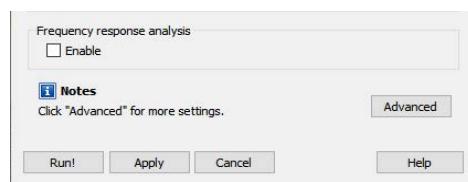
Parameter	Value
Minimum pitch, region 3 [rad]	0.0174533
Rated speed [rad/s]	121.6805
Rated power, region 3 [W]	5 296 610.0
Transition speed, region 1 and 1.5 [rad/s]	70.16224
Transition speed, region 1.5 and 2 [rad/s]	91.21091
Torque constant, region 2 [Nm/(rad/s) <sup>2</sup> ]	2.332287
Rated slip percentage, region 2.5 [%]	10.0
Maximum torque, region 3 [Nm]	47 402.91
Maximum torque rate [Nm/s]	15.0

## 3.5 Advanced solving and analysis

When the wind turbine model is completed, with proper blade definitions, control system, load conditions, aerodynamic setup, etc., the simulation can be started by a simple click of a button, as described in [Section 3.3, "Basic solving and analysis"](#). However, often the model is so complex that more tuning of the simulation setup is needed. We will describe some additional tools available in Fedem WindPower, for the solving and analysis of wind turbine models below.

### 3.5.1 Running the dynamics solver (advanced mode)

You can display the *advanced* Dynamics Solver Setup dialog box by choosing **Dynamics Solver (Advanced Mode)**... on the *Solve* menu, or click the **Advanced** button (shown to the right) in the lower right part of the *basic* Dynamics Solver Setup dialog box, discussed in [Section 3.3.1, "Running the dynamics solver \(basic mode\)"](#).



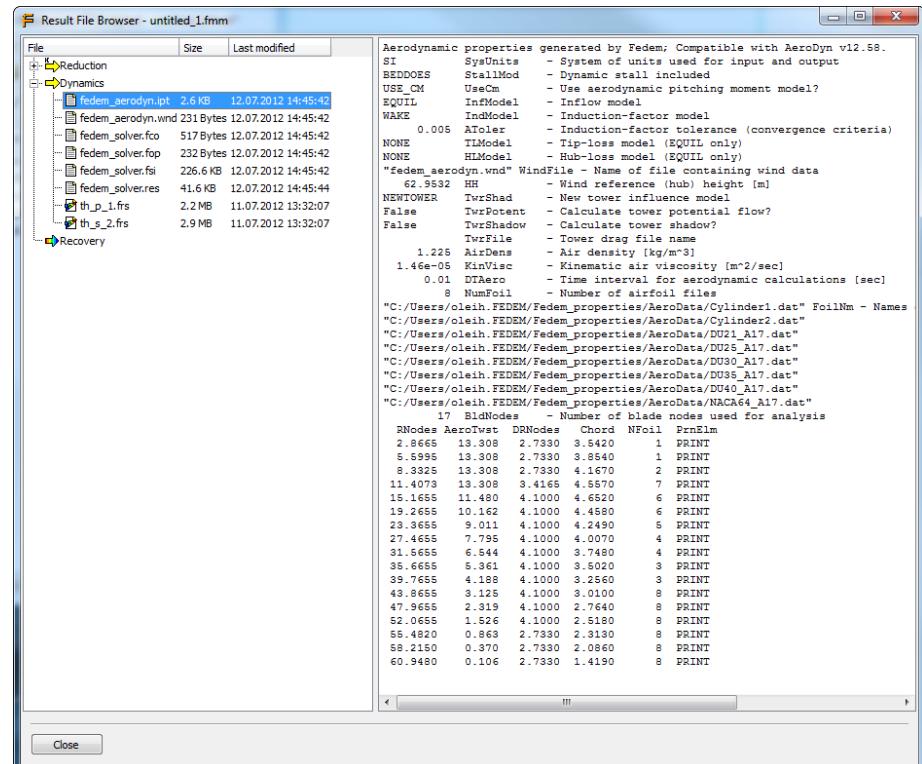
The advanced Dynamics Solver Setup dialog box contains several settings where you, for instance, can adjust the time integration parameters, nonlinear iterations, convergence tolerance settings, etc. Refer to [Section 8.5.2, "Dynamics Solver \(Advanced Mode\)"](#) for full details on all the options available in this dialog box.

### 3.5.2 Advanced results analysis

We will have a look at a few advanced analysis techniques here that are useful when analysing wind turbine models.

#### Result file browser

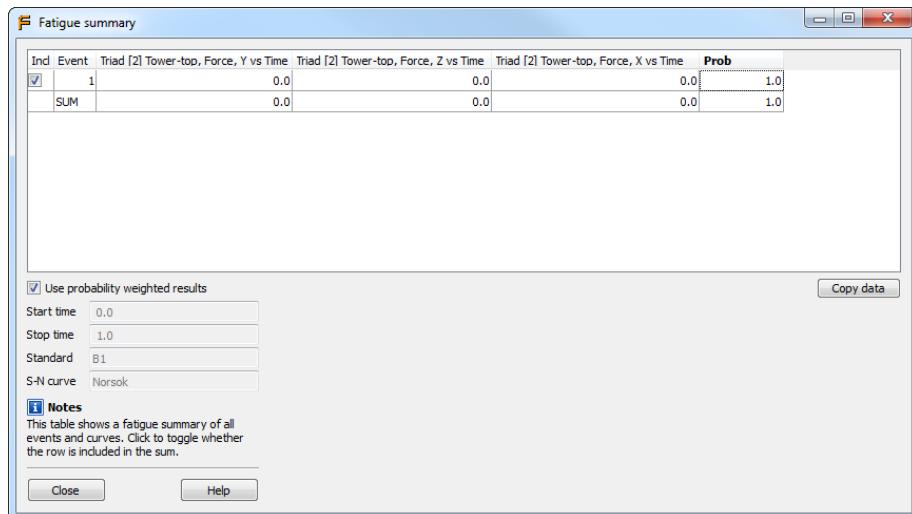
When running the dynamics solver, a results database is established on disk. The figure below illustrates the Result File browser, which is a dialog box for viewing the contents of the results database. See [Section 10.2, "Result File Browser"](#) for a general description of this dialog box. For wind turbine models, there is an additional set of files listed here one should know about:



First of all is the .ipt file, which is the main input file for AeroDyn. The content of this file can be viewed (as shown in the figure above) by clicking on the file name in the left part of the dialog box. Here you can see what is sent to AeroDyn. Next is the .wnd file, which contains the description of the wind field. This can either be a deterministic wind, or a turbulent wind, generated by TurbSim.

### Fatigue summary

Another useful tool is fatigue summary. You can do a fatigue summary analysis on any graph or set of curves in Fedem WindPower. Just select a graph or set of curves, right click them, and choose **Fatigue Summary...**. If you have multiple events in your model, then fatigue summary will be calculated for each event. You can adjust the probability values (Prob).



# Chapter 4 Mechanism Modeling

Now that you have been introduced to the graphical user interface of Fedem, you can start the detailed modeling process.

This chapter describes how to perform the various commands you need to build mechanism models, such as creating, moving, attaching, and detaching elements. It also describes how to apply motion constraints to the model.

The basic mechanism elements of Fedem (parts, joints, triads, functions, sensors, etc.) and their properties are discussed in detail in [Chapter 5, "Mechanism Elements"](#).

4

Sections in this chapter address the following topics:

- [Basic assembling techniques](#)
- [Mechanism modeling environment](#)
- [Mechanism modeling tools](#)
- [Creating mechanism elements](#)
- [Moving mechanism elements](#)
- [Attaching and detaching elements](#)
- [Deleting mechanism elements](#)
- [Using file references in mechanism elements](#)
- [Model preferences](#)

## 4.1 Basic assembling techniques

There are three main approaches to assemble a Fedem model.

- *Using modeling Add-ons* - You may create your own tailored modeling tool and load it into Fedem as a plug-in (see [Section 2.12, "Customizing Fedem using Addons"](#)), and then use this to generate a Fedem assembly directly.
- *With FE models or VRML geometry* - If you have VRML geometry files or FE-models of your parts, assembling a Fedem model means to import the parts, fit the parts together by moving and/or rotating them, and then connecting the parts by creating and attaching joints.
- *With hard-point positions* - When you only have hard-point information, it is more convenient to place the joints in space at the hard-points, and then connect the joints by creating generic parts or beams from the triads in each joint.

Other mechanism elements such as springs, dampers, loads, and so on can be added in the same manner, either by placing triads on an FE-model, or placing them in space, and attach them to generic parts.

When assembling a model in Fedem, triads are the system model counterpart of the FE-nodes and represent the connection between the system model and the parts. See [Section 5.5, "Triads"](#) and [Section 4.6, "Attaching and detaching elements"](#) for a description of how connections are made using triads.

As you assemble the model and move things around, Fedem tries to help you by setting the movability of objects to match the constraining of your model. This means that you will be unable to move an object (such as a ball joint) that is fixed in some way. If an object is constrained from translating, you will be able to rotate but not translate it, and so on.

## 4.2 Mechanism modeling environment

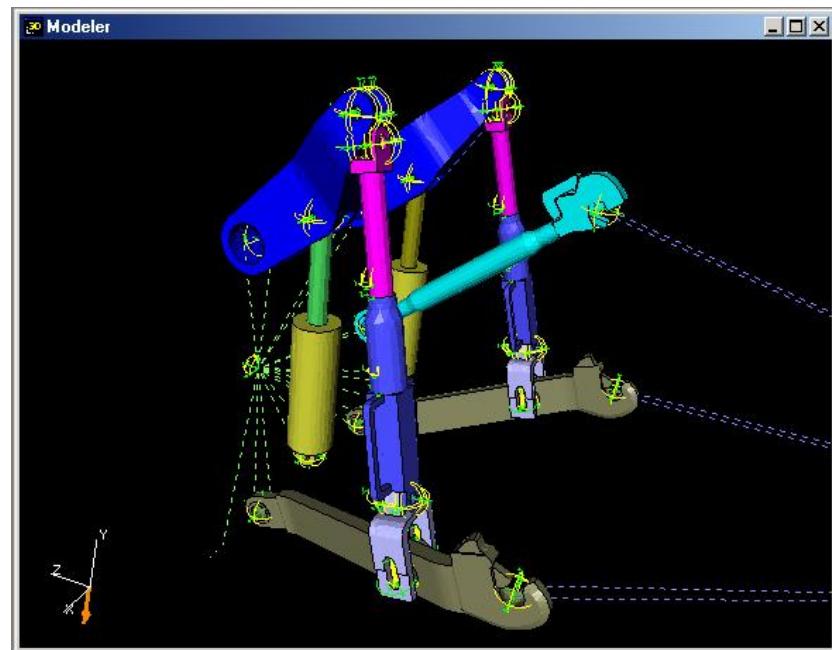
The Fedem modeling environment combines a powerful, 3D graphic interface and dynamic viewing capabilities with quick and easy management tools. The Model Manager tabs provide shortcuts for creating, selecting, and deleting elements.

### 4.2.1 Modeler view

To build a mechanism model, Fedem provides the *Modeler* view; a three-dimensional, graphical environment in which your model can be viewed and edited. The mechanism elements are selected from menus and tool bars for placement in the *Modeler* view. They can then be moved and manipulated using various modeling tools. This editing environment also features dynamic viewing tools, defined-view commands, and appearance settings (see [Section 2.6, "Visualizing the model"](#)).



To open the *Modeler* view, click the **Show Modeler** button on the Windows menu or tool bar. The *Modeler* view is shown below with an example mechanism assembly.



4

### 4.2.2 Modeling tool bars

In Fedem, there are three major tasks performed by the user: 1) creating the mechanism and control system; 2) setting up and starting the analysis; and 3) setting up and viewing the results. Each task has a different set of associated commands. The mechanism modeling tools used to create and edit models are covered by the *Mechanism Creation* tool bar and the *Mechanism Tools* tool bar.

### Mechanism Creation tool bar

The *Mechanism Creation* tool bar (shown below) contains the mechanical elements used to build Fedem mechanisms (see [Section 4.4, "Creating mechanism elements"](#) for instructions on using these commands, and [Chapter 5, "Mechanism Elements"](#) for a detailed description of each element).



**NOTE:** An arrow (▼) beside a button indicates that more options can be accessed by clicking and holding down the button.

### Mechanism Tools tool bar

The *Mechanism Tools* tool bar (shown below) consists of modeling tools. Each of these commands is described in the following sections.



## 4.3 Mechanism modeling tools

To help you position items with greater accuracy and to simplify the modeling process, Fedem provides some helpful modeling tools, including a reference plane, an interactive point locator, point markers, and movability constraints.

### 4.3.1 Reference Plane

The Reference Plane is the shaded area in the center of the *Modeler* view. It serves both as a visual reference, and as a representation of the ground.

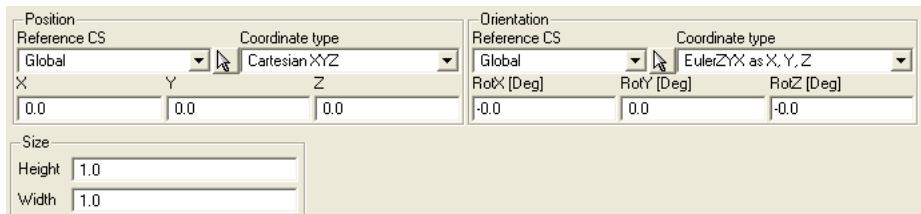
You can move it around, change its color and size, or turn it off so that it is not visible in the *Modeler* view.

To disable, enable or change the appearance of the Reference Plane, see [Section 2.6.5, "General Appearance"](#) and [Section 2.6.6, "Item Appearance"](#).

#### Changing the size

To change the size of the Reference Plane, select the Reference Plane in the *Modeler* view (or Model Manager *Objects* list), then edit its *Height* and

**Width** fields in the Property Editor panel (shown below). Remember to press **Enter** after typing the values to apply the changes.



### Moving

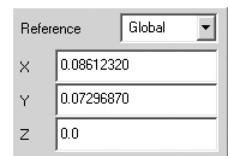
The Reference Plane is moved by editing the *Position* and *Orientation* fields in the Property Editor panel. See [Section 4.5.4, "Origin property"](#) for a detailed description of these data fields.

It is also possible to move the Reference Plane by aligning it to a specified coordinate system in your model. To do so, use the **Align CS** or **Align rotation** commands. See [Section 4.5.2, "Align CS and rotations"](#).

4

### 4.3.2 Interactive Odometer and 3D Point Marker

Many Fedem commands require you to select a point in your model. To help you locate specific points, Fedem provides the Interactive Odometer and the 3D Point Marker (shown at right). These are displayed in the *Modeler* view each time you select a point. The odometer shows the coordinates of the selected point, and the marker shows the location of the point in the *Modeler* view.



When using the **Smart Move** command to move or rotate parts and other mechanism elements (see [Section 4.5, "Moving mechanism elements"](#)), the Interactive Odometer allows you to edit the selected point or enter a new 3D point using global or local coordinates. The local coordinate system used is the coordinate system of the item you selected when the point was picked.



**TIP:** You can use the Interactive Odometer with the **Smart Move** command to place a mechanism element (part, joint, triad, and so on) at a point in free space. The object can then be used as a reference when moving other objects.

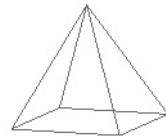
To edit a point or enter a new point using the Interactive Odometer, complete the following steps:

1. Select a point in the *Modeler* view. The coordinates of the point (given in the local coordinate system for the selected element) are displayed in the Interactive Odometer, and the 3D Point Marker shows the location of the point selected.
2. Select *Local* or *Global* coordinates from the *Reference* pull-down menu.
3. Type new values for the *X*, *Y*, and *Z* coordinates in the Interactive Odometer, and press the *Enter* key after editing the values. The 3D Point Marker is updated to show the new position.
4. When you are satisfied with the new location, press **Done** to confirm the selected point.

### 4.3.3 Stickers

Stickers are movability constraints that are applied automatically when moving mechanism objects with the **Smart Move** command (see [Section 4.5, "Moving mechanism elements"](#)). Stickers can also be applied manually (see ["Manually applying stickers"](#) below). Each sticker applies the same constraint as a ball joint; in other words, it constrains all translational motion.

Stickers are displayed in the *Modeler* view as small pyramids (shown at right). The tip of the pyramid is located at the constrained point.



When using **Smart Move**, the motion allowed, or movability, for a selected object or group depends on the number and location of applied stickers. Each move using **Smart Move** automatically applies an additional sticker. Therefore, three successive moves of an initially free object (without stickers) first results in a translation, then in rotation about a point, and lastly rotation about the remaining axis. The object is then locked in place. (See also ["Movability"](#) in [Section 4.5.1](#).)



**IMPORTANT!** Stickers function as modeling aids only. They are not considered part of the mechanism model, and therefore do not influence the mechanism motion during simulation.

#### Manually applying stickers

Stickers are created automatically when using **Smart Move**, but you can also create them manually when you need a certain type of movability. To rotate about a point in space, apply one sticker at the rotation center. To rotate about an axis, apply two stickers somewhere along the rotation

axis. Stickers are applied to Triads and Parts, and can be positioned at any point in space. They are not restricted to the geometry of visualization or the nodal points of a part. To create a sticker, perform the following steps:



1. Click the **Sticker** button on the *Mechanism Tools* tool bar. The Guide panel prompts you to select an application point for the sticker on an object.
2. Place the cursor over the point on the object you want and press the left mouse button. The selection snaps to the nearest node or point on the object.
3. If necessary, edit the position using the Interactive Odometer as described in [Section 4.3.2, "Interactive Odometer and 3D Point Marker"](#).
4. Confirm the point by clicking **Done**. The sticker is created, and the sticker symbol appears in the *Modeler* view at the selected point.

4

### Deleting stickers

You can delete stickers individually or all in a single operation.

- To delete a single sticker, complete the following steps:
  1. Select the sticker you want to delete in the *Modeler* view or from the Model Manager *Objects* list. The sticker symbol in the *Modeler* view turns red when selected.
  2. Click the **Delete** button on the *Standard* tool bar or use the **Delete** key. The sticker is removed from the model.
- To delete all the stickers applied to your model, click the **Delete All Stickers** button on the *Mechanism Tools* tool bar or in the *Mechanism* menu.



**NOTE:** You may have to click and hold down the **Sticker** button on the *Mechanism Tools* tool bar to access the **Delete All Stickers** command.



**WARNING!** There is no undo option after deleting all stickers. To replace them in your model, you must recreate each of them individually.

## 4.4 Creating mechanism elements

Objects such as Spring/Damper characteristics, Functions, Frictions, Beam- and material properties, which all do not need to be positioned are created in the Model Manager *Objects* list. Right-click an empty space in the Model Manager panel and select **Create** to access the full list of elements that can be created using the shortcut menu.



**TIP:** It is also possible to create non-positioned mechanism objects by copying existing ones. This is useful if you need a property object (Function, Friction, Beam property, etc), which is nearly identical to other existing property objects in the model. To copy objects, select them in the Model Manager Objects list, and then select **Copy Object** from the right-click menu, or from the Edit menu.

All mechanism elements that need to be positioned in the Fedem model are created by a different method. Normally, you first need to select the position(s) needed to place the new element. Then you sometimes need to orient the element properly, and then you finally have to attach it. See [Section 4.5, "Moving mechanism elements"](#) and [Section 4.6, "Attaching and detaching elements"](#).

The only exceptions to this are the parts and beams which are created completely differently. Please have a look at [Section 5.1.2, "Creating parts by file import"](#), [Section 5.1.3, "Creating parts from hard points"](#) and [Section 5.3.1, "Creating beams"](#).

To create a positioned mechanism element (except for parts and beams), complete the following steps:

1. Click the button for the item on the *Mechanism Creation* tool bar (see "[Mechanism Creation tool bar](#)" in [Section 4.2.2](#)).
2. Follow the instructions in the Guide panel while selecting one or more positions or related objects in the *Modeler* view. Positions can also be entered by using the Interactive Odometer. (See [Section 4.3.2, "Interactive Odometer and 3D Point Marker"](#).)
3. When you are satisfied with a selection, click **Done** to confirm. When all positions/selections are completed, The element is created using the selected position(s). A sticker is normally applied to the new object automatically to make it easy to rotate using **Smart Move**.
4. You can then use **Smart Move** or some other means to adjust the object's orientation (see [Section 4.3.3, "Stickers"](#) and [Section 4.5, "Moving mechanism elements"](#)).
5. As the last operation you need to attach the object to a part or a beam using the **Attach** command. See [Section 4.6, "Attaching and detaching elements"](#).



**TIP:** To edit the properties of a new mechanism element, select the item in the *Modeler* view or *Model Manager Objects* list. The properties of the item are then displayed in the *Property Editor* panel.

#### 4.4.1 Selecting position and orientation

When creating mechanism elements, you are asked to select their position. Revolute joints, Free joints, Loads and Cam joint master triads will be created with a default orientation as well.

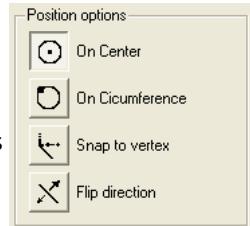
As you select a point, Fedem will snap to geometric features and also extract a default orientation if necessary, by different rules depending on what type of object you hit.

##### Snapping and default orientation on FE Parts

When picking an FE Element surface the default orientation is set to be perpendicular to that surface and the position snaps to the closest node. If the exact position of the mouse is on an FE mesh line, however, the default orientation is aligned with the direction of that line. It is the exact mouse position that decides whether you hit a line or a surface, even when the 3D Point Marker snaps to the same node.

##### Snapping and default orientation on CAD Parts

If the mouse position is on a CAD part, the geometry of the face or edge is used to extract a default orientation and a snap point. If the face or edge in question is a revolved geometry, the options shown to the right pop up in the Guide panel. These options allow you to control how the snap point and the default orientation is extracted from the geometry.



You can here choose whether the point shall snap to the center axis of the revolved face (*On Center*) with the default orientation along that axis, or to the circumference of the revolved geometry (*On Circumference*) with the default orientation perpendicular to the face.

The default orientation can be flipped to the opposite direction using the *Flip Direction* option.

The *Snap To Vertex* option controls whether or not the selected point shall snap to the closest vertex. In combination with *On Circumference*, the point will snap to the closest vertex on the surface, whereas with *On Center* it will snap to the point on the center axis which is closest to the vertex.

### Default direction notes

The default direction is visualized as a yellow arrow (as shown to the right) starting from the selected point. The following object types utilize the default direction:

- *Force and Torque* - the attack direction.
- *Revolute joint* - the axis of rotation.
- *Cam joint master triads* - the x-direction (up) of the masters are aligned with the default direction.
- *Free joint* - Z-Axis of the master.



## 4.5 Moving mechanism elements

Fedem provides several commands and tools for moving parts of your model: **Smart Move**, **Align CS**, **Align rotations** and **Move To Center** are useful commands while the *Origin* tab of the Property Editor panel, that several mechanism objects share, can be used to access the position and orientation of a single component directly.

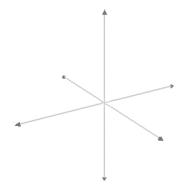
### 4.5.1 Smart Move

The **Smart Move** command is an integral part of 3D modeling in Fedem. It is a sophisticated way to position and orient mechanism entities such as parts, triads, joints, springs, dampers, and so on. This command enables you to move or orient an object or a group of objects according to the selection's current movability (see below). During a **Smart Move**, a movability constraint called a Sticker is automatically created (see [Section 4.3.3, "Stickers"](#)).

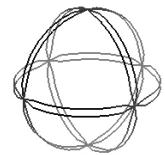
#### Movability

The movability of an object or a group of objects is determined by examining not only the stickers that are applied to the selection, but also the joints between the selected object/group and other objects. Each joint or sticker that constrains the group reduces its movability. The following symbols represent the six types of motion allowed when using the **Smart Move** command:

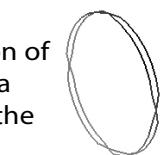
- **Free** – when an object without stickers or attached joints is moved with the **Smart Move** command, it can move in translation in any direction. The symbol for free movement is depicted in the *Modeler* view as shown to the right.



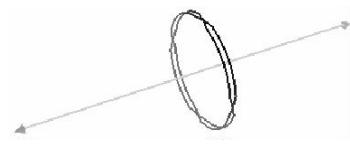
- **Ball** – when one sticker or one ball joint has been applied to a mechanism entity, it can rotate about the point at which the sticker/joint is applied. The symbol for ball movement is depicted in the *Modeler* view as shown to the right.



- **Revolving** – when two stickers (or one ball joint and one sticker, or two ball joints) have been applied to a selection of mechanism elements, the stickers/joints act together as a revolute joint with the axis defined by the line between the two stickers/joints. The symbol for revolving motion is depicted in the *Modeler* view as shown to the right.



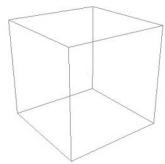
- **Cylindric** – when a selection of mechanism elements is constrained by one cylindrical joint, the selection can be translated along and rotated about the joint axis. The symbol for cylindrical motion is depicted in the *Modeler* view as shown to the right.



- **Prismatic** – when a selection of mechanism elements are constrained by one prismatic joint, the selection can be translated along the joint axis. The symbol for prismatic motion is depicted in the *Modeler* view as shown to the right.



- **Rigid** – when three stickers or ball joints (not located on a straight line) are applied to a selection of mechanism elements, the selection is fully constrained and cannot be moved with the **Smart Move** command. The symbol for rigidity (no movement allowed) is depicted in the *Modeler* view as shown to the right.



### Performing a Smart Move

To move an object or group of objects using the **Smart Move** command, complete the following steps:



1. Click the **Smart Move** button from the *Mechanism Tools* tool bar. The Guide panel prompts you to select objects to move.
2. To select an object and indicate the from-point, place the cursor over a point on the object and press the left mouse button. The selection snaps to the nearest node or point on the object. This point becomes the from-point and a symbol is shown that depicts the movability of the current selection.



**TIP:** Several objects can be selected by pressing and holding the **Ctrl** key while selecting objects. To change the last selected object only, release the **Ctrl** key and select until you hit the right object. To remove several of the last selected objects from the selection, release the **Ctrl** key, and press the left mouse button on some empty space in the Modeler view until all the objects in question is deselected.



**TIP:** You can also type in a discrete point or edit the point using the Interactive Odometer (see [Section 4.3.2, "Interactive Odometer and 3D Point Marker"](#)).

3. When you are satisfied with the object and the from-point selected, press **Done** to confirm it. The Guide panel then prompts for you to select a to-point.
4. Select the to-point in the same way you selected the from-point, and if necessary, edit it using the Interactive Odometer to specify a discrete point.
5. When you are satisfied with the to-point, press **Done** to confirm it. The move operation is animated in the *Modeler* view.

## 4.5.2 Align CS and rotations



Two align commands can be used to align one or several objects to an existing coordinate system in your model. The **Align CS** command will both translate and rotate the selected objects to match their coordinate systems (CS) with a selected object, while the **Align rotations** command only rotates the selected objects.

### Performing an Align command

To move an object or group of objects using one of the align commands, complete the following steps:

1. Click the appropriate align button in the *Mechanism Tools* tool bar. They are located under the **Smart Move** icon.
2. Select the objects to move by picking them in the *Modeler* view. Press **Done** to confirm the selection.



**TIP:** Several objects can be selected by pressing and holding the **Ctrl** key while selecting objects. To change the last selected object only, release the **Ctrl** key and select until you hit the right object. To remove several of the last selected objects from the selection, release the **Ctrl** key, and press the left mouse button on some empty space in the Modeler view until all the objects in question is deselected.

3. Select the coordinate system to align to by picking an object that is defined in that coordinate system. The selected coordinate system will be displayed in red. Press **Done** to confirm the selection and execute the move.



**TIP:** The **Align** commands can be used to align objects to local FE coordinate systems, if present. The visibility of local coordinate systems can be toggled using the General Appearance dialog box. See Section 2.6.5, "General Appearance".

### 4.5.3 Move To Center



This is a useful tool if you want to position an object at the center of some geometry. The object will move to the center of a circle you define, or somewhere along its axis. The new orientation of the object aligns with the circle. The x-axis is defined by the center and the first point defining the circle, the z-axis is perpendicular to the circle.

4

#### Performing a Move To Center

To move an object or group of objects using the **Move To Center** command, complete the following steps:

1. Click the **Move To Center** button in the *Mechanism Tools* tool bar. It is located under the **Smart Move** icon.
2. Select the objects to move by picking them in the model view. Press **Done** to confirm the selection.

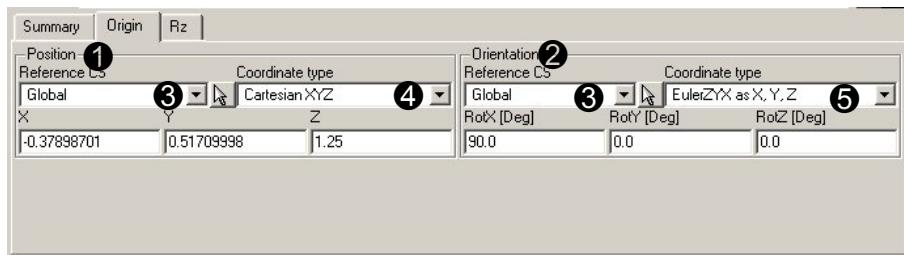


**TIP:** Several objects can be selected by pressing and holding the **Ctrl** key while selecting objects. To change the last selected object only, release the **Ctrl** key and select until you hit the right object. To remove several of the last selected objects from the selection, release the **Ctrl** key, and press the left mouse button on some empty space in the Modeler view until all the objects in question is deselected.

3. Select three points of the perimeter defining a circle. Confirm each point by pressing **Done**. After setting the third point the defined circle will appear in the Modeler view.
4. You may now select a point to place your objects along the circles axis, or press **Done** once more to move the objects to the center of the circle.

#### 4.5.4 Origin property

Triads, parts, and point-to-point joints have a property tab called *Origin* (shown below). This property tab is used to display and edit the position and orientation of the mechanism element in question. The sensitivity of the fields will reflect whether the selected object is allowed to move considering its attachments, etc., without corrupting the model.



- ① *Position* - This frame displays the data for the translational part of the position.
- ② *Orientation* - This frame displays the data for the rotational part of the position.
- ③ *Reference CS* - The translation and rotation can both be displayed and edited in any coordinate system in the model. These pull-down menus allow the reference coordinate systems to be selected. They can also be selected by picking in the *Modeler* view or selected from the *Objects* list of the Model Manager panel. To do so, you must first click the arrow button next to the pull-down menu. The Guide panel tells you to select a reference CS. Select a triad, a beam or a part, when satisfied, press **Done**.
- ④ *Coordinate type* - This menu controls whether to display the translation in Cartesian coordinates or cylindrical coordinates. The cylindrical coordinates can use either X, Y or Z as the rotational axis.
- ⑤ *Coordinate type* - This menu controls how to edit and display the orientation.

The Orientations input type options are:

- Angles about the X-, Y- and Z-axis in degrees. The parameterization used is the one called Euler-ZYX, which means a rotation about the Z-axis of the reference CS first, next the Y-axis, and finally the X-axis. (This can also be understood as a rotation about the axes of the with-rotated (or local) X-axis first, then local Y and finally local Z.)

- A point on the X-axis, and a point in the XY plane. The X and Y direction is then computed from the given point and the translation of the object.
- A point on the Z-axis, and a point in the XZ plane.
- A vector in the X direction and a vector in the XY plane. In this mode, the X and the Y vectors form a 3x3 rotation matrix that can be used directly.

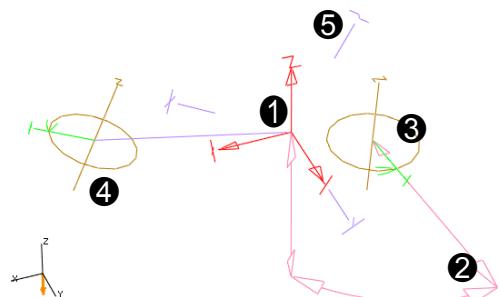


**NOTE:** The center of the applied rotations are always the origin of the coordinate system in question (the position controlled by the Position options) and not at the origin of the selected reference CS.

4

### Visualization

The sizes displayed in the Position frame are visualized, along with the reference CS for the orientation, whenever the *Origin* property is visible. The visual appearance of this visualization is shown to the right.



- ① *The object to move (red)* - In this example; a Triad.
- ② *Position arrows (pink)* - Shows the various dimensions corresponding to the selected Coordinate Type as arrows extending from the reference CS for the position.
- ③ *Reference CS for the position*.
- ④ *Reference CS for the Orientation* - The reference CS for the Orientation is shown by a purple line from the object to move to the reference CS.
- ⑤ *Orientation Reference direction* - The reference directions for the orientation are shown as purple lines and letters indicating the orientation of the reference superimposed on the position of the object to move.

## 4.6 Attaching and detaching elements

Nearly all mechanism elements created in Fedem needs to be attached to a part or a beam (or two parts or beams). The concept of attaching is to connect the joint constraints, loads etc. to the parts/beams they affect.

When attaching, two things happen: Firstly a triad in the element in question is noted to be connected to the part/beam (see also [Section 5.5, "Triads"](#)). All the constraints loads etc. that the element introduces will then be working on that particular part or beam.

Secondly, if attaching to an FE part, the triad is connected to the FE mesh of the part, either by direct association with an existing FE node in the part, or by using a *Surface connector* to distribute the forces in some way.

When an element is attached, it can generally not be moved relatively to the object it is attached to. The **Detach** command is used to disconnect a mechanism element from the part or beam it is attached to, making it possible to move it around again.

### 4.6.1 Using the Attach command

The **Attach** command can be used when an element is to be connected to ground, to a beam, to a *Generic Part*, or to an *FE part* with an existing FE-node at the attach point. To attach a mechanism element using the **Attach** command, complete the following steps:



1. Click the **Attach** button on the *Mechanism Tools* tool bar (or select from the *Mechanism* menu). The Guide panel prompts you to select a mechanism element to attach to the model.
2. Select the element in the *Modeler* view.
3. When you have made your selection, press **Done** to confirm it. The Guide panel then prompts you to select a part (or beam) onto which to attach the object.
4. Select a part, a beam, or the reference plane in the *Modeler* view and press **Done** to confirm the selection. The object becomes attached to the selected part or beam, or to the ground if the reference plane was selected.



**TIP:** Watch the Output List view for error messages during the attachment process. If an attachment cannot be completed, an error message is displayed here.



**NOTE:** Triads in axial springs, dampers, and loads are automatically attached when created, because the orientation of such triads is not important.

## Attaching joints

Joints consists of one slave triad and one or more master triads. The slave triad is normally attached to one FE-model, and the master triad(s) to another. This is done by attaching one part of the joint first, and then the other one. See [Section 5.6, "Joints"](#) for more information about master and slave triads in joints. When attaching by selecting the joints directly, Fedem automatically selects which triad (master or slave) to attach first. In most cases, the slave is attached first.

To control whether a joint's slave or master is attached to a specific node, select only the part of the joint symbol that represents that particular triad when selecting the object to attach during the **Attach** command. See [Section 5.6, "Joints"](#) for more information about joint symbols.



**TIP:** To attach multiple joints to a single FE node, you must align the master triads of the joints. You can then attach the master triads of each joint to the FE node. The two master triads will then be merged into one triad shared by the two joints.

4

### 4.6.2 Surface Connectors



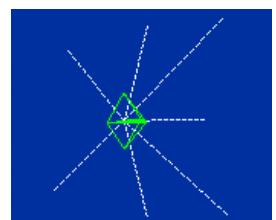
The **Surface connector** commands are used to attach mechanism elements to FE parts at positions without existing FE-nodes (e.g., at center of holes), or in such a way that the Surface connector distributes the forces from the joint, load etc. onto some area on the FE model.

Surface connectors connects a triad to an FE model using two different connection types, *Rigid Surface* or *Flexible Surface*.



#### Flexible surface

The flexible surface connector acts as a force distributor. It does not introduce stiffness or constraints between the FE nodes it connects to, but distributes the forces from the triad onto the FE model.



Each nodal DOF gets an equal share of the translational forces at the triad. The moments acting on the Triad, and the moment created by the translational forces about the geometrical center of the nodes, is balanced by an additional force in each node, weighted by the nodes distance from the geometrical center of the nodes in the Surface connector. In the case where the nodes also have rotational DOFs (shell nodes) those rotational DOFs also gets an equal share of the moment.

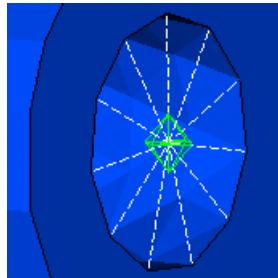
The Flexible surface connector is visualized with dotted lines.



### Rigid surface

The rigid surface connector is a rigid connection between the triad and all the FE-nodes it connects. That means that all the FE-nodes connected with the surface connector becomes one rigid block.

The rigid surface connector is visualized with dashed lines.



### 4.6.3 Surface connector commands

There are two ways of creating a Surface connector: **By selecting nodes** or **By cylinder surface**.

The commands can either create a new connected triad at a user defined position, or connect an existing triad to the FE mesh, in somewhat the same manner as the **Attach** command.



#### By selecting nodes

These commands are used to select arbitrary nodes or areas of nodes to use for the surface connector.

To create a Surface connector by selecting nodes, complete the following steps:

1. Select either the rigid or flexible version of the command.



2. First you have the option to either select an existing triad, which might be embedded in a mechanism element, or to select a position where you want a new triad to be created.

If you pick a triad, that triad will become selected and will be attached through the surface connector. If you pick something else, the snap-point will be used as the position for a new triad.

If you get it wrong, try again until you have selected what you want, then accept by pressing **Done**.

3. Now you need to select all the FE nodes to be connected to the triad. You can add single nodes to the selection by picking, or all visible surface nodes within a rectangle by dragging a window. If you press and hold the **Ctrl** key, picked or window selected nodes will be removed from the selection instead of being added.

When finished, press **Done**.



### By cylinder surface

These two commands creates a Surface connector from nodes on the surface of a cylinder volume. It is convenient to use them to attach a mechanism element to the inside of a hole, a circular edge etc. The command is also able to place a new triad along the axis of the cylinder, making it easy to get the hard-points. This triad can then be used for further modeling.

4

The cylinder is defined by a 3-point circle, together with points on each end of the cylinder.

To create a Surface connector by cylinder surface, complete the following steps:

1. Select either the rigid or flexible version of the command.



2. You have the option to select either an existing triad that will be attached by the Surface connector, or a position where a new triad will be created. If you want the command to create a triad along the cylinder axis, start by selecting the tree nodes that define the circle.

If you pick a triad, that triad will be attached through the surface connector. If you pick an FE-node, the node will be used as the first of three nodes that defines the cylinder circle. If you pick something else, the snap-point will be used as the position for a new triad.

If you get it wrong, try again until you have selected what you want, then accept by pressing **Done**.

3. Then select the rest of the tree points defining the cylinder circle, accepting each node by pressing **Done**. A cylinder/circle visualization will show the resulting cylinder as you select the last node.

4. When the cylinder circle is defined, you can now or after any of the following steps press **Done** to complete the command using the definition of the cylinder that is shown. The optionally new triad will then be created in the center of the circle, if you did not define a position for it in the start of the command.
5. The next steps is to select the start and end of the cylinder. Do this by selecting an FE-node for the start, and one for the end. Accept each of them by pressing **Done**.
6. Finally the position of the optionally new triad along the cylinder axis can be selected. Press **Done** to accept.

When the command is completed, Fedem selects all the nodes from the FE-model in question that is on the surface of the cylinder volume defined. These nodes is now connected to the new or existing triad by the Surface connector.

#### **Deleting or redefining Surface connectors**

A surface connector is actually an attribute of the Triad it connects. This means that if the triad is detached, the connector is deleted.

If a connector needs to be edited or changed, simply use one of the surface connector commands to redefine it. Invoke the command, and select the triad with the mis-defined connector at the start of the command sequence.

#### **4.6.4 Attachment rules and restrictions**

There are several restrictions and rules that apply to the connections between FE models and triads. These restrictions do not apply when attaching triads to *Generic Parts* or *Beams*.

- The triad and the FE node must be within the distance set by the modeling tolerance (see [Section 4.9.2, "Modeling tolerance"](#)). If several nodes exist within the modeling tolerance, the one closest to the triad will be selected.
- Slave triads can not be attached to ground. Joints must be attached to the ground by their master triads.
- Two or more slave triads can not be attached to the same FE node.
- Master triads or triads where the triad directions is referenced must be aligned before they can be attached to the same FE node.



**NOTE:** When several elements are attached to the same node, the triads in those elements are merged into the triad already attached. This resulting triad is then shared by all the attached elements. (The Redundant triads are removed.)



**NOTE:** When a triad is attached to a FE model at the position of a slave FE-node, Fedem will automatically add a 6-DOF node, a spring and a mass element to the FE model at that point. The triad is then attached to the new 6-DOF node instead of the slave node to overcome limitations in the mathematical methods used. The spring stiffness and mass is set automatically by the Reducer to a very high stiffness, and a very low mass compared to the actual model in question. However, if the part is completely rigid (e.g., it consists of a single RGD element), the value  $2e11$  is used for stiffness and no mass is added. See also the Fedem 7.6 Theory Guide, Section A.10 "BUSH" and Section A.12 "CMASS".

## 4.6.5 Detaching

To detach mechanism elements from your model, complete the following steps:



1. Click the **Detach** button on the *Mechanism Tools* tool bar (or select from the *Mechanism* menu).
2. Select the item(s) to detach (hold down the **Ctrl** key while selecting multiple items in the *Modeler* view).
3. Click **Done** to confirm your selection. The object(s) are detached from your model.



**NOTE:** When detaching a joint, both the master and slave triads are detached. If you want to detach only one of them, press the **Detach** button, then select the part of the joint symbol that represents either the master or the slave triad and press **Done**.

## 4.6.6 Color of attached and unattached elements

The color of a mechanism element indicates whether or not it is attached to the model. If the element's symbol is white (the default color), it is not fully attached. A colored symbol indicates that an element is attached.



**TIP:** The colors for attached and unattached elements can be changed in the General Appearance dialog box (see [Section 2.6.5, "General Appearance"](#)).

## 4.6.7 Invalid attachments

At some points Fedem may find that some Triads in your model does not correspond to an underlying FE-node even if it should have. Fedem will then warn you with a dialog box, and mark all the invalidly attached triads with a red exclamation mark in the Model Manager Objects list.



The model will not be solvable as long as you have invalidly attached triads. To resolve this you have several options:

- Use a surface connector to connect the triad to a set of existing FE nodes.
- You can move the triads to the correct positions.
- Update the FE-mesh with nodes at the correct positions.
- Turn the FE-part into a Generic Part.
- Delete the Triads.

## 4.7 Deleting mechanism elements

Fedem uses the **Delete** command to remove mechanism elements from a model. The **Delete** command can be used in two different ways; in the *Modeler* view or in the Model Manager panel.

### 4.7.1 Deleting in the Modeler view

To delete elements in the *Modeler* view, complete the following steps:

1. In the *Modeler* view, select the element to be deleted, or hold down the **Ctrl** key and select multiple items. The selected items are highlighted in red.
2. To remove the selected element(s) from the model, click the **Delete** button on the *Standard* tool bar or hit the **Delete** key.



**WARNING!** There is no undo option after deleting objects. To replace mechanism elements after deleting them, you must recreate each of them individually.

### 4.7.2 Deleting in the Model Manager panel

To delete mechanism elements in the Model Manager panel, complete the following steps:

1. In the *Objects* list, select the item to be deleted, or hold down the **Shift** or **Ctrl** key and select multiple items. The selected items are highlighted in red in the *Modeler* view.



**TIP:** You can also click and drag the cursor to select multiple items for deletion.

2. To remove the element(s) from the model, right-click and select **Delete** from the shortcut menu (or hit the **Delete** key). The items are removed from both the model and the list at the same time.



**NOTE:** When deleting parts and beams, you have the option to retain triads that are attached to each deleted part/beam (except for joint triads - they are always retained), to also delete those triads, or to cancel the **Delete** command. If no such triads exist for a selected part or beam, you must anyway confirm the deletion of the part/beam. This choice must be made for each part and beam in the selection. However, by selecting the **Yes to all**, **No to all** or **Ok to all** button, you automatically repeat the same choice for each part and beam in the current selection.



**NOTE:** If any of the objects you delete are referred to by a curve, you will be able to choose to either delete that specific curve definition, leave the curve definition intact while still deleting the object, or cancel deletion of the selected object.



**WARNING!** If the deletion of a selected object also causes deletion of other objects connected to it, and any of these other objects also are referred to by a curve, you will be notified and can choose whether to delete that specific curve definition or not, too. However, you can not choose to cancel the Delete operation at this stage.

## 4.8 Using file references in mechanism elements



Some mechanism elements in Fedem need input from external files. For such elements the use of file references may be beneficial. The file reference replaces the file name in the element definition. As the contents of the file reference, the file it is referring to, is changed, so is the element input. Thus, if several mechanism elements receive their input from the same file reference, and the contents of the file reference changes, so does the input of all elements using it.

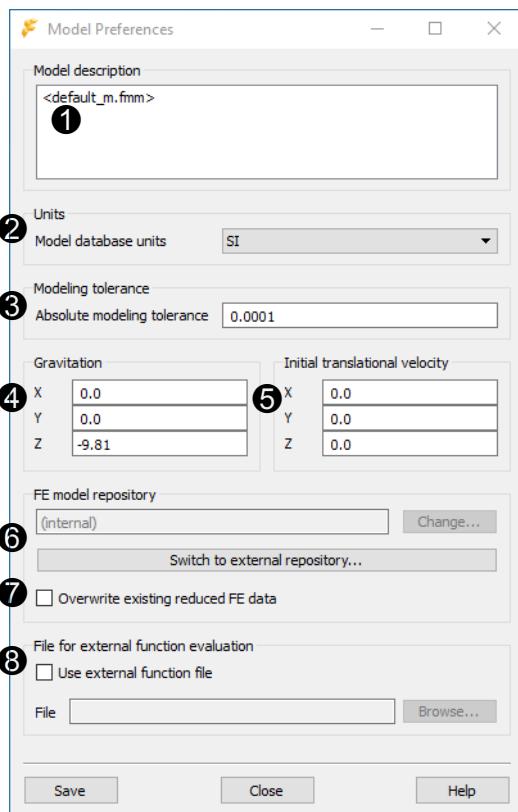
File references are created either by choosing **File reference...** from the *Mechanism* menu, or by right-clicking an empty space in the *Objects* list of the Model Manager panel, choosing **Create** and then **File reference...**. Then select one or more files in the dialog box that appears. One File reference object will be created for each file selected.

File references are set as input in mechanism elements by choosing the wanted reference from the pull-down menu of the input file field.

## 4.9 Model preferences

In the Model Preferences dialog box you can adjust several global parameters regarding the model.

- ① **Model description** - This field lets you add notes to your model file.
- ② **Units** - The consistent unit set to use. See [Section 4.9.1, "Model database units"](#).
- ③ **Modeling tolerance** - The maximum allowed distance between the FE-node and its attached Triad. See [Section 4.9.2, "Modeling tolerance"](#).
- ④ **Gravitation** - The gravity vector direction and magnitude. See [Section 4.9.3, "Gravitation"](#).
- ⑤ **Initial translational velocity** - The initial translational velocity of all triads in the model. See [Section 4.9.4, "Initial translational velocity"](#).
- ⑥ **FE model repository** - Switch between external and internal model repositories. You may also **Change** the external FE model repository. See "[Setting the FE model repository](#)" in [Section 5.1.6](#).
- ⑦ **Overwrite existing reduced FE data** - If enabled, any existing data in the FE model repository for an FE part that is reduced will be overwritten, instead of creating a new parallel sub-folder for it. Use this if your model consists of large FE parts that are reduced several times during modeling sessions, in order to save disk space.
- ⑧ **File for external function evaluation** - If enabled, you can browse for a file from which the [External functions](#) in the model will take their values, instead of expecting them from an outside process. See [Section 4.9.5, "External function values from file"](#).





**NOTE:** Changes to the Model description notes will be saved also when closing the Model Preferences dialog box (using the **Close** button). Therefore, you may edit these notes at any time, also when you have results.

### 4.9.1 Model database units

All modeling in Fedem is unit independent. However, some external interfaces may require SI units in their calculations.

When modeling in other units than SI, you will then need to change the model database units. Do this by selecting the model database units corresponding to your model. The chosen units are then used to properly scale calculated data before communicating with the external modules that require a specific unit set.



**TIP:** You can add your own modeling units by editing the file `units.fcd` in any ASCII editor. This file is located in the Fedem installation directory.

4

### 4.9.2 Modeling tolerance

The modeling tolerance is a tolerance that controls how strict Fedem enforces that triads and their corresponding FE nodes are coincident. This tolerance needs to be strict, because an offset will introduce an error and inconsistency in the numerical model which might have serious impact on the reliability of the results.

The seriousness is however depending on the size of the offset compared to the size of the FE models in question and the overall size of the model. If you experience problems when trying to attach triads to a part or beam, you might need to increase the modeling tolerance.

The default tolerance is set to  $10^{-4}$  of the length unit you are using. That is a good tolerance when using meters as model database length unit, but is probably too strict when using millimeters. You will have to adjust this tolerance to some sensible value according to the size of your FE models and the units you work in.

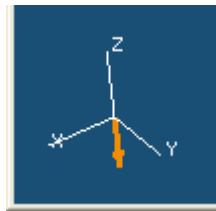


**WARNING!** When decreasing the tolerance in a model that is built using a large modeling tolerance, some of the triads/joints might become invalidly attached when reopening the model.

### 4.9.3 Gravitation

The gravitation size and direction can be adjusted in the Model Preferences dialog box. Remember to edit this in order to correspond to the units you are using.

The direction of the gravitation vector is displayed by the orange arrow in the lower left corner of the *Modeler* view (shown at right).



### 4.9.4 Initial translational velocity

The complete mechanism can be given a uniform initial translational velocity, by entering a velocity vector in the Model Preferences dialog box. This velocity is then distributed to all Triad DOFs in the model, except for those that has been assigned a non-zero initial velocity in the Property Editor panel (see "[Free DOF](#)" and "[Prescribed DOF](#)" in [Section 5.5.3](#)).

This is useful if your event actually is describing the mechanism moving at some speed different from zero. If a non-uniform initial velocity state is defined (by entering different values for appropriate triads and/or joint DOFs), care must be taken such that the velocity state specified is consistent throughout the model. If that is not the case, severe fictitious transients may occur in the first time steps of the simulation.

### 4.9.5 External function values from file

When developing a mechanism model as a *Digital twin* of a real asset, the use of external functions is unavoidable (see "[External function](#)" in [Section 5.11.5](#)) for feeding the sensor readings from the asset into the digital twin model. However, numerical simulation of such a model will not work unless connected to the system delivering the sensor readings. Therefore, you can specify a file from which the external function values will be taken when run as a stand-alone process. This is useful when developing the model or performing some basic test simulations.

The specified file needs to have one column for each external function in the model, and each column needs to be labeled with the *Tag* of the Function that should use that column. The columns can either be comma-separated (csv-file), or separated by one or more white-space characters. In the latter case, in the first line containing the column labels (also white-space separated), the first column must be labeled "#*Description*". For csv-files, the first column label is arbitrary.

The external function values file must contain one line of data for each simulation time step. The first column (which normally contains the time), is not in use. If the file is too short, i.e., it contains fewer lines of data than simulation time steps, the final line of data will define the external function values for the remaining time steps of the simulation.



**NOTE:** *The data in the external function values file will not be interpolated, if the time steps of the simulation do not match the times in the file (this in contrast to if the same file is used in a "Polyline from file" function).*



# Chapter 5 Mechanism Elements

Now that you know how to create and assemble mechanism elements, you need to know how Fedem defines the properties of elements and how you can customize them to suit your design requirements. This chapter presents each of the mechanical and modeling elements used in Fedem mechanisms. It also describes each element's properties and how they can be edited once the element is created.

Sections in this chapter address the following topics:

- [Parts](#)
- [Element groups](#)
- [Beams](#)
- [Beam cross sections](#)
- [Triads](#)
- [Joints](#)
- [Joint pair constraints](#)
- [Frictions](#)
- [Springs and Dampers](#)
- [Loads](#)
- [Functions](#)
- [Sensors](#)
- [Strain rosettes](#)
- [Generic database objects](#)
- [Sub-assemblies](#)

## 5.1 Parts

As described in [Chapter 1, "Introduction to Fedem"](#), parts are the basic components of Fedem models. You can connect parts with various types of joints to create a movable *mechanism*. Each part is either an *FE Part* represented by a standard FE model , or a *Generic Part* represented by a simplified model forming a semi-rigid connection between other mechanism entities.

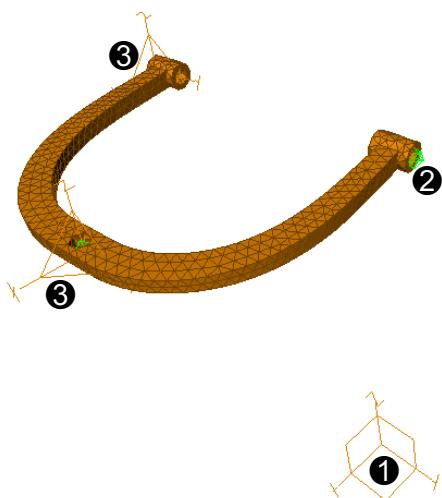
Two-noded beams may also be used in place of *Parts* when constructing a Fedem model. A beam is somewhat equivalent to a Generic Part connected to two triads, except that the mass- and stiffness matrices are derived consistently from a linear beam finite element, instead of using the semi-rigid representation. Such elements are denoted *Beams* in the *Model Manager* and are described in [Section 5.3, "Beams"](#).

### FE Parts

The mass-, stiffness- and dynamic properties of an FE Part are defined through its FE model, defined by nodes, elements, materials and physical property data. The FE model must be constructed in an external FE modeler, such as one of those described in [Section 1.5, "Using FE models"](#), and then imported into Fedem.

You can use simple or complex parts in your models, depending on your needs and modeling capabilities. Shown below is a simple FE part modeled with solid elements.

- ① *Part coordinate system* – The FE model representing the part is defined in the part coordinate system. When the part is imported into Fedem this coordinate system is aligned with the global coordinate system.
- ② *Triads (optional)* – nodal points that are defined as external during the construction of an FE model in an external modeler, are connected to automatically generated triads when the part is imported into Fedem.



If external nodes are not defined in the FE model file, no triads will appear when the model is imported (see [Section 5.5, "Triads"](#) for more information about triads).

- ③ *Local coordinate systems* – Local FE coordinate systems present in the imported FE model is read and displayed for reference.

### Generic Parts

A Generic Part is a simplified flexible body. It is purely defined by its connection points, mass properties and stiffness at the connection points. The stiffness can either be defined manually, or automatically set to some very high value, mimicking a rigid body. See the Fedem 7.6 Theory Guide, [Section A.16 "Generic Part element"](#), for details on how a Generic Part is represented in the Dynamics Solver.

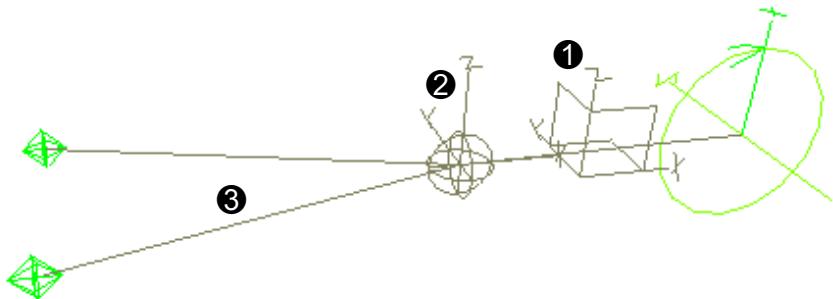


**TIP:** As an alternative to using Generic Parts, entirely user-defined elements may also included in your model. User-defined element types can be programmed according to a defined interface, and loaded into Fedem as a plugin. See [Section 2.11.2, "User-defined elements"](#) on how to implement your own set of user-defined elements.

5

Generic Parts can be used when you have no FE model for the part, when trying to optimize hard-point positions, or when the flexibility of the part is considered to be insignificant. They can also have a VRML geometry attached, to give better visualization of the part.

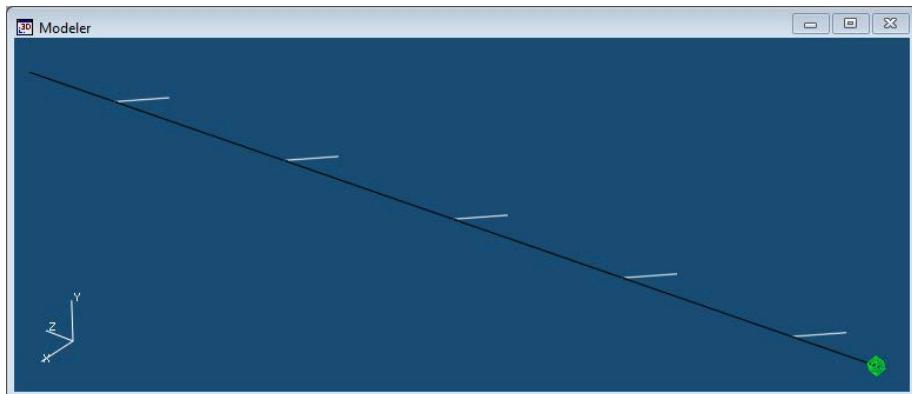
Shown below is a Generic Part with two triads and a Revolute Joint connected to it.



- ① *Part coordinate system* - At the time of creation, the part coordinate system is in the global origin.
- ② *Centre of Gravity* - The Generic Part's centre of gravity can be positioned independently from the part coordinate system.
- ③ *Simplified visualization* - The lines extending from the centre of gravity to each of the triads attached provides a coarse visualization of the Generic Part.

### 5.1.1 Visualization of beam element orientation and eccentricity in FE Parts

For FE Parts containing beam elements, each beam element is represented by a straight line connecting its two end nodes in the Fedem Modeler view. This is sufficient for elements with a circular cross section where the properties will be independent of the element orientation. However, for all non-circular cross sections some visualization of element orientations is required to verify that the properties are correctly defined. A full rendering of the actual cross section will provide such verification, but this may result in a too big visualization model for FE Parts containing a lot of beam elements. Instead, Fedem provides a simplified visualization of the beam element orientation.



The image above shows a simple FE part consisting of 5 beam elements and with element orientation visualization activated. It consists of a white line in the local XZ-plane of each beam element, extending from the mid-point of each element towards local end 1 of the element such that it forms a 45-degree angle with the local X-axis of the element. The length of the orientation line is constant and independent of the element length itself. Its length is scaled by the factor entered in the *Size* field in the *General Appearance* dialog box, see [Section 2.6.5, "General Appearance"](#).

The beam element orientation visualization is switched off by default. To enable it, activate the *Local coordinate systems* toggle in the *General Appearance* dialog box, see [Section 2.6.5, "General Appearance"](#).

Beam elements may also have an eccentric coupling to the FE nodes. The beam axis is in this case visualized in its eccentric position, and with a white line connecting the eccentric beam end with the associated FE node.

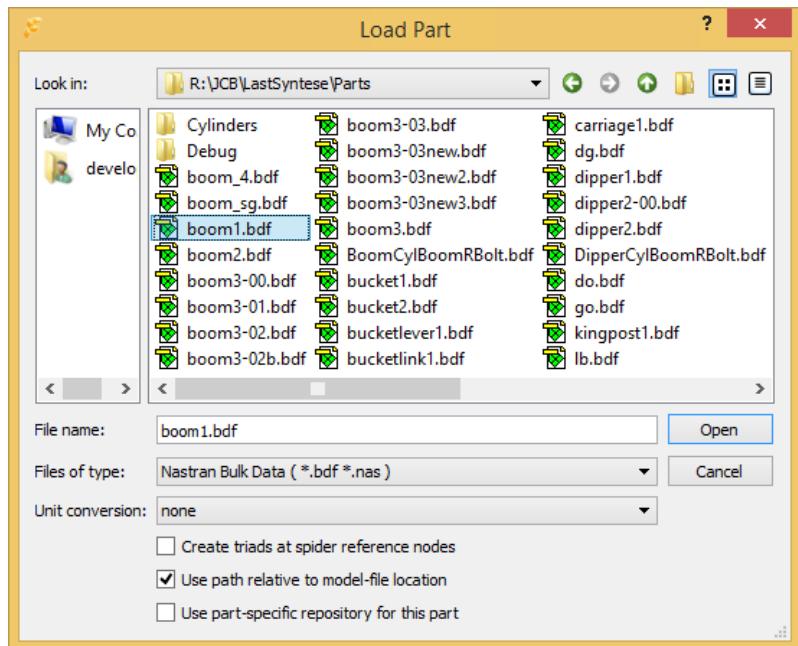
## 5.1.2 Creating parts by file import

Parts can be created by importing FE model files or by importing CAD geometry as VRML files. The available file formats are listed in [Section 2.2, "Storing models and results"](#).

Importing FE models will create an FE Part, while importing a VRML file will create a Generic Part. In any case, complete the following steps, to import a part:



1. Click the **Load Part** button on the *Mechanism Creation* tool bar (or select from the *File* menu). The dialog box shown below then opens.



2. Select the file type you want from the *File of type* pull-down menu.
3. Browse for or enter the path and name of the file in the *File Name* box.



**TIP:** You can import more than one part at the same time by holding down the **Ctrl** or **Shift** key and selecting multiple files in the Load Part dialog box.

4. Select the units conversion you need from the *Unit conversion* pull-down menu. All units used in the file for dimensions and properties are converted according to your selection.



**WARNING!** There is no connection between this unit conversion and the model database units. You must be careful to choose the conversion that fits your needs.

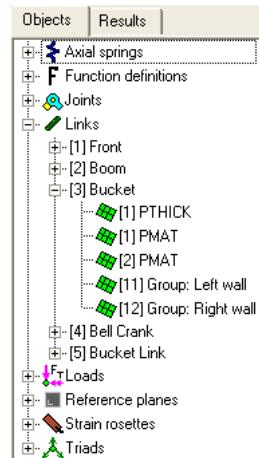


**TIP:** You can add your own unit conversions by editing the file `units.fcd` in any ASCII text editor. This file is located in the Fedem installation directory.

5. If the FE model to be imported contains “spider” elements, i.e., elements of types RGD or WAVGM (see [Section , "Fedem Technology Link format"](#)) or the Nastran elements RBE2 or RBE3 (see [Section A.1.8, "Nastran Bulk Data File format"](#)), you may automatically create Triads at the reference nodes of these elements by enabling the toggle *Create triads at spider references nodes*.
6. Make sure that you want to store the relative path to the original FE model or the VRML model. This setting is relevant when copying and moving the model across file systems, and has most impact on VRML files. The path to the original FE model file is only used if the internal FE model copies are lost.
7. Indicate if you want this part to be a part of the FE model repository, or if it should be stored with a part-specific repository. See [Section 5.1.6, "Using FE model repositories"](#).
8. Once you have selected the file(s) and a unit conversion, click **Open**. The part files are imported, and the coordinate system of the parts are aligned with the global coordinate system.

When an FE model is imported, it is converted to the internal Fedem format. During this conversion, several element groups might be created as well. These element groups can either be user-defined explicit groups, or implicit groups based on the properties of the elements in the FE part.

All element groups associated with a FE part are listed in the *Objects* list of the Model Manager panel, as illustrated to the right. See [Section 5.2, "Element groups"](#) to learn more about element groups in Fedem.



### 5.1.3 Creating parts from hard points

If you only have the hard point information of a part, you can create a part from triads positioned at the hard points as follows:

1. Select the triads that represents the hard points of the part using the multi-select features of Fedem. See [Section 2.4.3, "Model Manager"](#) and [Section 2.5.1, "Select"](#) on how to select multiple objects either from the Modeler view or in the Model Manager Objects list.



2. Click the **Create Generic Part** button on the *Mechanism Creation* tool bar. A Generic Part will be created with its origin in the global origin, and with its centre of gravity in the geometric centre of the triads you selected.
3. Set up the centre of gravity and mass properties of the part.
4. Optionally add more triads to the part by attaching them.



**TIP:** You can at any time during modeling attach or detach new triads to the Generic Part. See [Section 4.6, "Attaching and detaching elements"](#).

#### 5.1.4 Copying parts

If you need to use the same FE model or Generic Part more than once, you can duplicate an existing part by completing the following steps:

1. Select the part you want to copy from the *Modeler* view (or Model Manager Objects list).
2. On the *Edit* menu, select **Copy Part**.



The new part is placed offset from the original.



**TIP:** You can also copy parts using the shortcut menu in the Model Manager Objects list. Right-click the part you want to copy and select **Copy Part** from the shortcut menu. The new part is placed offset from the original in the Modeler view.

#### 5.1.5 Part properties

Parts are the basic components of any Fedem mechanism. It is therefore essential to understand the part properties that are displayed in the Property Editor panel when you select a part.



**TIP:** In addition to the settings found in the Property Editor panel, you may also find some information on the part's underlying FE model by selecting the part in the Result File Browser (see [Section 10.2.1, "The Result File Browser dialog"](#)).

The part properties are separated into several tabs to better organize the different settings. The number of tabs and their content depend on whether the part is defined as a Generic Part or an FE Part, or if it is a part used for visualization only (grounded parts).

The different tabs are as follows:

- **Part tab** - Always present
- **Origin tab** - Always present
- **FE Node tab** - Present for FE Parts (not for grounded parts)
- **Reduction Options tab** - Present for FE Parts (not for grounded parts)
- **Reduced Loads tab** - Present for FE Parts, if element or nodal point loads are present in the FE data file (not for grounded parts)
- **Mass tab** - Present for Generic Parts only (not for grounded parts)
- **Stiffness tab** - Present for Generic Parts only (not for grounded parts)
- **CoG tab** - Present for Generic Parts only (not for grounded parts)
- **Hydrodynamics tab** - Always present unless the part is grounded
- **Advanced tab** - Always present unless the part is grounded

### Part tab

The *Part* tab displays some basic settings and information about the part. The actual options that are displayed depend on whether the part is a Generic Part, an FE Part or if it is used for visualization only. Below, three versions of this panel are displayed.

The figure consists of three vertically stacked screenshots of a software interface for managing parts. Each screenshot shows a tab bar at the top with 'Part', 'Origin', 'FE Node', 'Reduction Options', and 'Advanced'. The first two screenshots also have tabs for 'Mass', 'Stiffness', 'CoG', and 'Hydrodynamics'.

- Top Screenshot (FE Part):**
  - ①:** Radio button for 'FE Part' is selected.
  - ②:** Check box for 'Visualization only' is unselected.
  - ③:** 'Finite Element Model' section shows 'Repository entry' as '[internal] C:\Users\l331884\Fedem-src\solverTests\SubModelling' and 'Imported file' as '01\_01\_Global\_Coarse.nas'. A 'Change...' button is next to the repository entry, and 'No unit conversion' is checked.
  - ④:** 'Needs reduction' button is present.
  - ⑤:** 'Structural Damping' section contains fields for 'Mass proportional' (0.0) and 'Stiffness proportional' (0.0).
  - ⑥:** 'Scaling of Dynamic Properties' section contains fields for 'Stiffness' (1.0) and 'Mass' (1.0).
- Middle Screenshot (Generic Part):**
  - ①:** Radio button for 'Generic Part' is selected.
  - ②:** Check box for 'Visualization only' is unselected.
  - ③:** 'Finite Element Model' section shows 'Repository entry' as '[internal] C:\Users\l331884\Fedem-src\solverTests\SubModelling' and 'Imported file' as '01\_01\_Global\_Coarse.nas'. A 'Change...' button is next to the repository entry, and 'No unit conversion' is checked.
  - ④:** 'Needs reduction' button is present.
  - ⑤:** 'Structural Damping' section contains fields for 'Mass proportional' (0.0) and 'Stiffness proportional' (0.0).
  - ⑥:** 'Scaling of Dynamic Properties' section contains fields for 'Stiffness' (1.0) and 'Mass' (1.0).
  - ⑦:** 'Visualization' section contains a 'File:' field with an empty value and a 'Change...' button.
- Bottom Screenshot (Visualization only):**
  - ①:** Radio button for 'FE Part' is selected.
  - ②:** Check box for 'Visualization only' is selected.
  - ③:** 'Finite Element Model' section shows 'Repository entry' as '[internal] C:\Users\l331884\Fedem-src\solverTests\SubModelling' and 'Imported file' as '01\_01\_Global\_Coarse.nas'. A 'Change...' button is next to the repository entry, and 'No unit conversion' is checked.
  - ④:** 'Needs reduction' button is present.
  - ⑤:** 'Structural Damping' section contains fields for 'Mass proportional' (0.0) and 'Stiffness proportional' (0.0).
  - ⑥:** 'Scaling of Dynamic Properties' section contains fields for 'Stiffness' (1.0) and 'Mass' (1.0).
  - Message:** 'Part is ignored in solvers. Triads will be attached to ground' is displayed at the bottom.

- ① **FE Part/Generic Part** - Part type selector. You can switch between the FE model or the Generic Part model at any time during modeling.



**NOTE:** When switching between FE Part and Generic Part, Fedem tries to use the supplied FE data and visualization data in a sensible way. If your part is defined as an FE Part and you switch to Generic Part, the FE model is retained in memory and used as visualization unless a VRML model file is specified. Fedem will not drop the FE data from memory until you actively enter or change the VRML model file name. When switching back to FE Part, Fedem will check that all the Triads are validly attached to the FE model. See also [Section 4.6.7, "Invalid attachments"](#).

- ② **Visualization only** - You can define the part to be used for visualization only. The part will then be ignored by the solvers and actually serve as an extension of the *Reference plane* (see [Section 4.3.1, "Reference Plane"](#)). The part and all triads attached to it will thus be grounded.



**NOTE:** You can toggle a part as Visualization only at any time during modeling, but it is disallowed if the part has slave triads attached.

- ③ **Finite Element Model (FE Parts only)** - This group of options concerns the FE model used.

- *Repository entry* indicates the repository type and the name of the selected .ftl file in the Fedem FE part repository (for details on the FE model repository, refer to [Section 5.1.6, "Using FE model repositories"](#)). If the model file has not yet been saved, this entry will state the file name Fedem will use when you save your model.
- *Imported file* indicates the name and location of the originally imported FE model file. The unit conversion that was applied during the import is also displayed.
- The **Change...** button allows you to replace the FE model of the part with a new one. This button opens a dialog box in which the new part file can be chosen. Note that all mechanism entities attached to the part, that do not have corresponding nodal points in the new FE model, will be detached.

- ④ **Needs reduction (FE Parts only)** - This label is a flag that signals if your FE model has been reduced or not. If some data for the reduced part is present and is recognized to match the part, this entry will read *Reduced [n]*, where "n" is a number referring to the directory in the part database containing the reduced matrices.

- ⑤ **Structural Damping** – Allows you to change the values of both the mass and stiffness proportional damping for the part. These parameters are described in the Fedem 7.6 Theory Guide, *Section 7.5, "Structural damping"*.

- ⑥ Scaling of dynamic properties** – Allows you to scale stiffness and mass of each individual part in the dynamics simulation. The scaling is done during initialization and stays in effect for the entire analysis. This option is useful for sensitivity studies of deflection and stiffness.



**WARNING!** The mass and stiffness scaling is not accounted for in any of the recovery analyses, and the recovered result will thus be misleading on FE parts using mass and/or stiffness scaling.



**WARNING!** The damping matrix and the associated force vector are not affected by the mass- and stiffness scaling parameters. That is, when using mass- and/or stiffness-proportional damping, it is the unscaled mass and stiffness matrix that contributes to the damping matrix and force vector.



**WARNING!** The mass- and stiffness scaling is not accounted for during FE part reduction. Therefore, the component mode shapes are always computed from the unscaled mass and stiffness matrix. Using stiffness and/or mass scaling on an FE part having component modes might thus yield incorrect results unless the two scaling factors are equal (because the component modes then are computed from a different set of matrices than the one used in the dynamics simulation).

- ⑦ Visualization (Generic Parts only)** - This frame contains options and information regarding the visualization of the Generic Part. The **File** field can contain a path to a VRML file to use as a visualization for the Generic Part. Press the **Change...** button to browse for a file. If a valid FE model file is already referenced, Fedem will use that as a visualization until a VRML model file name is entered.

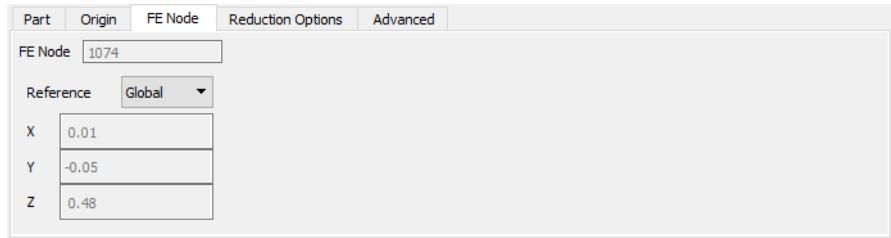
### Origin tab

The *Origin* tab is used to display and edit the position and orientation of the part. See [Section 4.5.4, "Origin property"](#) for a description of the data fields in this tab.

Part	Origin	FE Node	Reduction Options	Advanced																								
<table border="1"> <thead> <tr> <th colspan="3">Position</th> <th colspan="3">Orientation</th> </tr> <tr> <th>Reference CS</th> <th>Coordinate type</th> <th></th> <th>Reference CS</th> <th>Coordinate type</th> <th></th> </tr> </thead> <tbody> <tr> <td>Global</td> <td>Cartesian XYZ</td> <td></td> <td>Global</td> <td>Euler ZYX as X, Y, Z</td> <td></td> </tr> <tr> <td>X 0.0</td> <td>Y 0.0</td> <td>Z 0.0</td> <td>RotX [Deg] 0.0</td> <td>RotY [Deg] 0.0</td> <td>RotZ [Deg] 0.0</td> </tr> </tbody> </table>					Position			Orientation			Reference CS	Coordinate type		Reference CS	Coordinate type		Global	Cartesian XYZ		Global	Euler ZYX as X, Y, Z		X 0.0	Y 0.0	Z 0.0	RotX [Deg] 0.0	RotY [Deg] 0.0	RotZ [Deg] 0.0
Position			Orientation																									
Reference CS	Coordinate type		Reference CS	Coordinate type																								
Global	Cartesian XYZ		Global	Euler ZYX as X, Y, Z																								
X 0.0	Y 0.0	Z 0.0	RotX [Deg] 0.0	RotY [Deg] 0.0	RotZ [Deg] 0.0																							

## FE Node tab

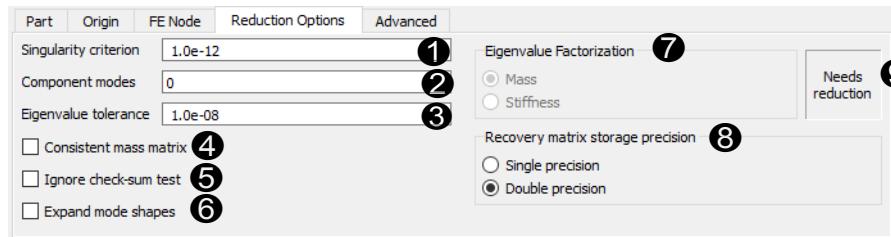
The *FE Node* tab is used to display information on FE nodes in parts. If you click with the left mouse button anywhere on an FE part in *Modeler* view, the 3D Point Marker  will snap to the closest node and the node number and coordinates are displayed in the Property Editor panel.



## Reduction Options tab

The settings on the *Reduction Options* tab affect how the part is reduced.

5



- ➊ *Singularity criterion* - This is the tolerance used to decide whether the stiffness and mass matrices are singular when they are factorized during model reduction. See "[Singularity tolerance](#)" in [Section 8.3.4](#).
- ➋ *Component modes* - Allows you to specify the number of component modes representing the internal (eliminated) nodal DOFs after CMS model reduction. See [Section 8.3.2, "Using component modes"](#).
- ➌ *Eigenvalue tolerance* - This is the maximum acceptable relative error in the computed eigenvalues in the fixed boundary eigenvalue analysis.
- ➍ *Consistent mass matrix* - Enables the use of consistent mass matrix in the model reduction process. If disabled, a lumped mass matrix approach is used. See [Section 8.3.3, "Using lumped mass matrix"](#).
- ➎ *Ignore check-sum test* - Disables the check on whether the reduced FE data, found in the FE model repository, is consistent with the part file currently used. Due to some rare numerical inconsistencies between reduced file data and the read FE data file, Fedem may signal that a part file needs reduction even though the reduced data are present.



**CAUTION:** Do not enable the *Ignore check-sum test toggle* unless you are sure that the reduced FE data found on disk are compatible with the current model. The consequence of using incompatible FE data may be a diverging model or incorrect results. A warning is issued whenever this toggle is enabled to stress this.

- ⑥ *Expand mode shapes* - Enables the expansion of component mode shapes and free-free mode shapes of the reduced part, for subsequent visualization. See [Section 8.3.6, "Visualization of eigenmode shapes from the model reduction"](#) and [Section 9.5, "Animations"](#).
- ⑦ *Eigenvalue Factorization* - Allows you to specify which matrix to be Choleski-factorized during the eigenvalue analysis that is performed in the component modes computation. Default is the mass matrix.
- ⑧ *Recovery matrix storage precision* - Allows you to switch to *Single precision* storage of the recovery matrix (a.k.a. the B-matrix) on disk. This will reduce the needed disk space for this matrix by 50%, and might be advantageous for very large parts with many triads that will result in a big B-matrix. The default is to use *Double precision* storage.
- ⑨ *Reduced/Needs reduction* - See "[Part tab](#)" above.



**CAUTION:** Switching to single precision storage of the B-matrix should normally have no influence on the dynamics simulation results. However, if the part's FE model is poorly conditioned (e.g., there is a large span in the stiffness properties over the part) there might be minor loss of accuracy in the recovery results due to the truncation of the B-matrix elements stored on file.



**NOTE:** For parts that are reduced with component modes (see bullet ② above), the single/double precision storage option also applies to the file containing the component mode shapes (the E-matrix file).

### Reduced Loads tab

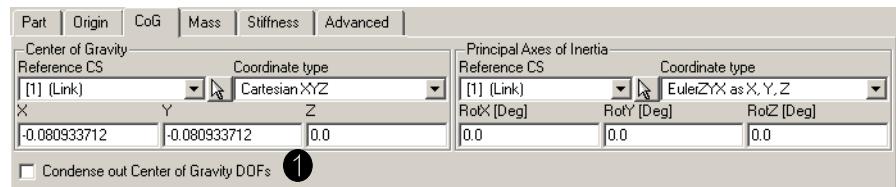
On this tab you can assign time histories for load cases that are defined in the FE data file. The associated reduced consistent load vectors are computed by the Reducer and stored in the FE model repository together with the reduced stiffness and mass matrices.

Load Case	Delay	Load Amplitude
1	0.0	2.3
2	0.0	[1] Load amplitude (Function)

- ❶ **Load Case** - This column contains the user-defined load case ID for each load set defined in the FE data file.
- ❷ **Delay** - If the load amplitude is defined by a Function, this value is used as a shift to the function argument, i.e., if the amplitude is specified as a function of time,  $f(t)$ , then the actual amplitude becomes  $f(t-Delay)$ . This is useful when each load case define the load state at different times in a transient simulation. The *Delay* can then be set equal to the time where each load case is active, and the same function can then be applied to each load case to obtain a smooth transition from one load case to the next.
- ❸ **Load Amplitude** - You can either enter a constant value or select a Function (or Time history input file) from the pull-down menu. This value or function will then be used as a scaling of the reduced load vector associated with this load case in the dynamics simulation.

### CoG tab

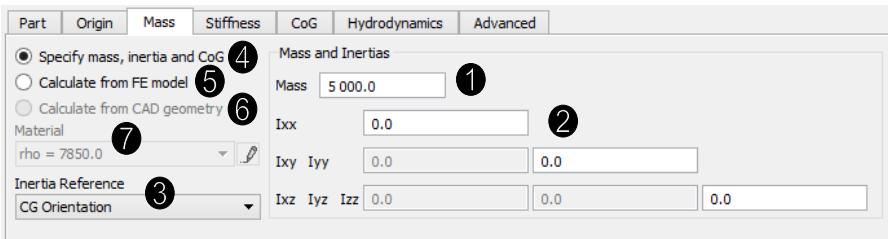
On this tab you can edit the position of the centre of gravity for a Generic Part. You can also enter the orientation of the principal axis of inertia to be used as the reference for the inertias entered on the [Mass tab](#).



- ❶ This toggle enables the elimination of the DOFs associated with the centre of gravity in the dynamics simulation. It is used to remove potential artificial internal vibrations in the Generic Part, and thus increase the numerical stability of the model.

### Mass tab

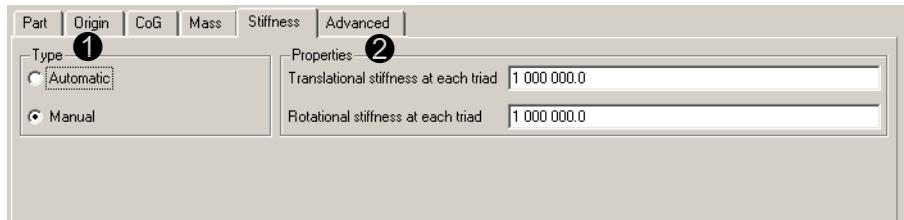
The settings on this tab concern the mass and inertia properties of a Generic Part, and is used to establish the part's mass matrix (see the Fedem 7.6 Theory Guide, Section A.16 "Generic Part element").



- ① *Mass* - The total mass of the part.
- ② *Ixx - Izz* - Lower triangle of the inertia matrix at the centre of gravity.
- ③ *Inertia Reference* - Select whether to specify the inertia in the directions of the part coordinate system or in the directions specified as *Principal Axes of Inertia* on the [CoG tab](#).
- ④ *Specify mass, inertia and CoG* - The part mass and inertias must be specified explicitly in the *Mass and Inertias* fields of this tab. The centre of gravity must also be specified in the [CoG tab](#).
- ⑤ *Calculate from FE model* - The part mass and inertias, as well as the centre of gravity in the [CoG tab](#), will be calculated from the FE model associated with the part. The *Inertia Reference* will also be set to *Part Orientation* and editing of the *Mass and Inertia* fields are disabled.
- ⑥ *Calculate from CAD geometry* - The mass, inertias, and the centre of gravity of the part will be calculated from the *Visualization* file associated with it, and the chosen *Material*. The *Inertia Reference* will also be set to *Part Orientation* and editing of the *Mass and Inertia* fields are disabled.
- ⑦ *Material* - If the *Calculate from CAD geometry* toggle is enabled, you can select the material object from which the mass density to be used in the mass property calculation is taken. If no material object is selected, the value 7850.0 is used (appropriate for steel materials).

## Stiffness tab

The stiffness properties of a Generic Part can be controlled on this tab.



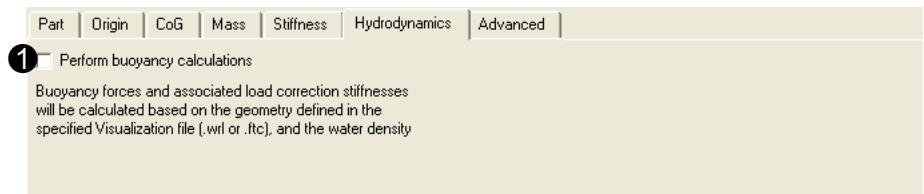
- ① **Type** - These radio buttons choose whether to set the overall Generic Part stiffness manually, or to have Fedem calculate a near rigid stiffness for you.
- ② **Properties** - This frame contains the stiffness values for the manually selected stiffnesses.

When setting the stiffness calculations to *Automatic*, Fedem uses the mass and a high target eigen frequency to calculate a sensible high stiffness. This will work as long as the mass of the part is set to something sensible. Thus Fedem is not able to calculate a good stiffness for a part with no mass at all, or with a mass that does not correspond to the actual use of the part. In such cases you will need to set the stiffness manually.

See the Fedem 7.6 Theory Guide, *Section A.16, “Generic part element”* for details on how the Dynamics Solver derives a part stiffness matrix from the manually specified stiffness values.

## Hydrodynamics tab

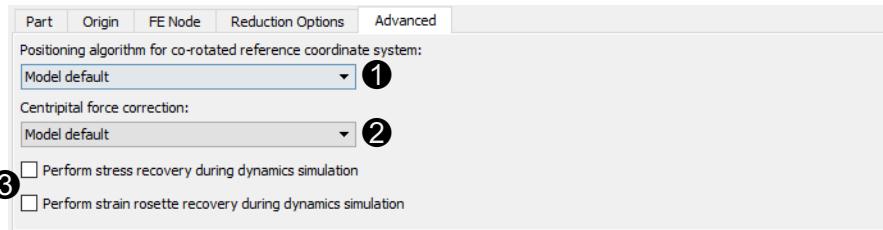
This tab concerns the calculation of hydrodynamic forces on a part.



- ① **Perform buoyancy calculation** - Enables the calculation of buoyancy forces for this part, provided a geometry description file has been specified in the *Visualization* field of the *Part tab*. This geometry file has to define a closed volume that represents the total displaced fluid body when the part is submerged. It can be either on the VRML-format, or the Fedem Technology Cad format (.ftc).

## Advanced tab

This tab contains drop-down menus and toggles for selection of some advanced settings that will apply during the dynamics simulation for the selected part.



- ① The options for the *co-rotated reference coordinate system* are:
  - *Model default* - The global setting defined in the [Integration tab](#) of the Dynamics Solver Setup is used (see [Section 8.5.2, "Dynamics Solver \(Advanced Mode\)"](#)).
  - *Max triangle, with unit offset when necessary* - This is the original algorithm, the only one available in Fedem R3.1.1 and earlier.
  - *Max triangle, with scaled offset when necessary* - The same as above, but with adjustments of the offset to better fit the part size.
  - *Mass based nodal average* - Algorithm based on equilibrium of a rigid shadow element with averaged stiffnesses at the triads.
 See the Fedem 7.6 Theory Guide, [Section 4.1, "Superelement local coordinate system"](#) for a detailed description of these algorithms.
- ② The options for the *centripetal force correction* are:
  - *Model default* - The global setting defined in the [Integration tab](#) of the Dynamics Solver Setup is used (see [Section 8.5.2, "Dynamics Solver \(Advanced Mode\)"](#)).
  - *On* - Turns centripetal force correction on for this part.
  - *Off* - Turns centripetal force correction off for this part.
- ③ These two toggles enable/disable the calculation of internal deformations and stresses/strains in, respectively, the finite elements and strain rosette elements in the FE part during the dynamics simulation, see [Section 8.5.6, "Stress and strain recovery during time integration"](#).

### 5.1.6 Using FE model repositories

A FE model repository is a directory structure containing all the files related to one or more FE parts. This includes the FE model files, reducer input option files, the reduced matrix and load files, the displacement recovery matrix files, and log-files with text-based output from the model reduction processes.

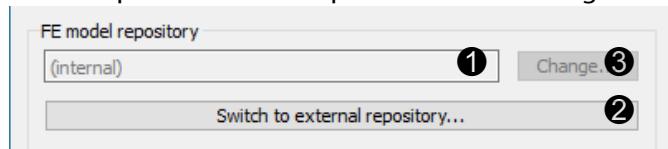
The term part database is also used when referring to a FE model repository. For detailed information about part databases, see [Section 10.3.1, "Part database"](#).

Fedem can handle FE model repositories in three different ways:

- *Internal repository* (default) – The default FE model repository is placed inside the model results database in the `Link_DB/` directory. This repository will follow the model, and be copied along with the results if saving the model as a new name.
- *External repository* – Sometimes it is useful to be able to share, and reuse the FE model repository among several model files. Using an external repository enables such sharing. If this is set, using *Save As* will not copy the FE model repository, but the original and the new model will point to the same repository and thus share any identical information.
- *External single part repository* – It is possible to use a specific FE model repository for an individual part. This is used to import and reuse a part from an existing FE model repository.

#### Setting the FE model repository

The FE model repository can be set in the Model Preferences dialog box (see [Section 4.9, "Model preferences"](#)). Select **Model Preferences...** from the *Edit* menu to open it. Below is a portion of that dialog box shown.



- ① The position of the current FE model repository.
- ② This button switches between internal and external repository.
- ③ The **Change...** button changes the external repository to a new directory. This might be an empty directory, or a directory used as FE model repository by another model.

When changing the repository, Fedem will copy the active content of the current repository to the new destination prompting on whether to overwrite if necessary. The old repository will be left untouched, unless it is an internal repository. Internal repositories will be deleted when switching to an external repository.

If there exists files with the same names at the new destination, Fedem will try to find out whether the reduced data match the existing FE model. If it does the reduced data will not be copied, as identical data is assumed to exist at the new destination.

### Reusing a part from an existing FE model repository

To import and reuse the reduced data for a part, the part can be imported from the existing FE model repository using the **Load Part...** command. See [Section 5.1.2, "Creating parts by file import"](#). Select the *.ftl* file you want, and toggle on the *Use part-specific repository* toggle in the Load Part dialog box. You will then need to add triads and reducer options for that particular part, that match the options and triad positions used by the previously reduced part. When done, Fedem will detect the reduced files, and flag the part as *Reduced*.

## 5.2 Element groups

When a part is created by importing a FE model into Fedem, several element groups might be created as well, see [Section 5.1.2, "Creating parts by file import"](#). The element groups can be of the following two types:

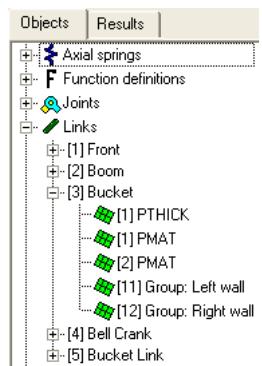
- **Explicit groups** User-defined group through Nastran SETs on bulk data files or Fedem GROUPs on *.ftl* files.
- **Implicit groups** An implicit element group consists of all elements referring to one particular material (PMAT) or thickness (PTHICK) property record in the FE model file. The ID numbers of these groups correspond to the ID numbers of the associated property record.  
Implicit groups are created only for property records that are in use.



**NOTE:** When the imported FE model file is a Nastran bulk data file, the created PMAT and PTHICK groups correspond to the MAT1 and PSHELL bulk entries, respectively, with corresponding ID numbers. No implicit groups are created for PSOLID bulk entries.

The element groups are visible in *Objects* list of the Model Manager panel under each Part node, as shown to the right. They can be used to control component appearance (see [Section 2.6, "Visualizing the model"](#)), calculation focus (see [Section 8.2, "Part- and group-wise solving"](#)).

The element groups are also used to assign properties needed in Fatigue analyses during Strain Coat Recovery simulations (see [Section 5.2.1, "Element group properties"](#) below, and [Section 8.9, "Strain coat analysis"](#)).



Some Nastran bulk data files may also contain a user-defined name of an element set or physical property as a comment line before the set/property definition itself. When found, such comments are parsed and used in the default description of the created element group when the bulk data file is imported into Fedem.



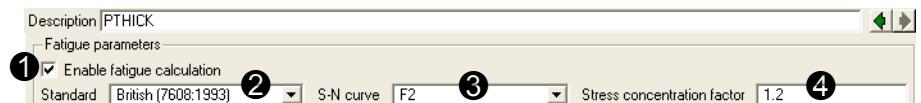
**NOTE:** The syntax of the comment lines containing names on element sets and properties depends on the software package that produced the actual Nastran bulk data file. Currently the syntax of the following packages are supported: I-DEAS, Hypermesh and NX.



**TIP:** The description field can be edited both for explicit and implicit element groups. The modified description is then stored in the part file (.ftl file). To revert to the original description (e.g. PTHICK for implicit groups based on the thickness element property), delete the description text completely. Any user-defined name parsed from a Nastran bulk data comment line is not restored, however.

## 5.2.1 Element group properties

When an element group is selected in the Model Manager panel, its properties are displayed in the Property Editor panel (shown below). It contains parameters and settings that are used in Fatigue calculations during a Strain Coat analysis on this element group. See [Section 8.9, "Strain coat analysis"](#) to learn more about such fatigue analyses.



- ① This toggle enables rainflow analysis and fatigue calculations for the selected element group.
- ② *Standard* - Select the fatigue standard to use for in the fatigue calculations.
- ③ *S-N curve* - Select an S-N curve from the selected fatigue standard.
- ④ *Stress concentration factor* - The computed stresses are scaled by this value before they are used in the fatigue calculation.



**TIP:** The S-N curve standards available in the pull-down menu are defined in the file *sn\_curves.fsn* located in the installation directory of Fedem. The syntax of the S-N curve definitions is description in the header of this file, and it is possible to add your own S-N curve definitions to that file.

For details on how damage is calculated from a given time history response, see the Fedem 7.6 Theory Guide.

## 5.3 Beams

A beam is a flexible body, represented by a standard two-noded beam finite element. Its stiffness matrix is based on Euler-Bernoulli beam theory, quadratic interpolation functions and continuous rotations at the nodal points. The deformations in such elements account for bending, shear, axial compression and elongation, and St. Venants torsion. The cross section properties are assumed constant along the beam element.

Beam elements may be used in a Fedem mechanism model, in much the same way as a generic part connected to two triads. A triad needs to be placed at each end of a beam element. However, an arbitrary number of beam elements may be connected at one triad, such that it is possible to represent a curved (or straight) beam string and 3D space frames.

### 5.3.1 Creating beams

To manually create one or a set of beam elements, proceed as follows:

1. Create Triads where you want to locate the nodes of the beam(s).
2. Select two or more triads that should represent the nodes of the beams using the multi-select features of Fedem (see [Section 2.4.3, "Model Manager"](#) and [Section 2.5.1, "Select"](#)).
3. Click the **Create Beam** button on the *Mechanism Creation* tool bar, or select it from the *Mechanism* menu.



A set of  $(nT-1)$  beam elements is then created, where  $nT$  denotes the number of selected triads. When three or more triads are selected, the second triad will become the second node of the first element created and also the first node of the second element. The third triad will next become the second node of the second element and the first node of the third element, and so on. This can therefore be utilized to create strings of beam elements of any shape and length.

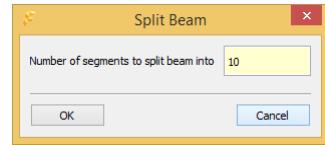


**NOTE:** The newly created beam elements have no cross section properties assigned. This needs to be done from the Property Editor panel when one or a group of beams is selected (see [Section 5.3.3, "Beam properties"](#)), before the simulation is started.

## 5.3.2 Splitting beams

An existing beam element may be divided into an arbitrary number of beams of uniform length, by using the **Split Beam...** command:

1. Select the beam you want to split from the *Modeler* view (or the *Objects* list of the Model Manager panel).
2. On the *Edit* menu (or the right-click menu), select **Split Beam....**
3. A small dialog box then pops up (shown to the right), in which you may enter the number of elements that you want the selected element divided into.



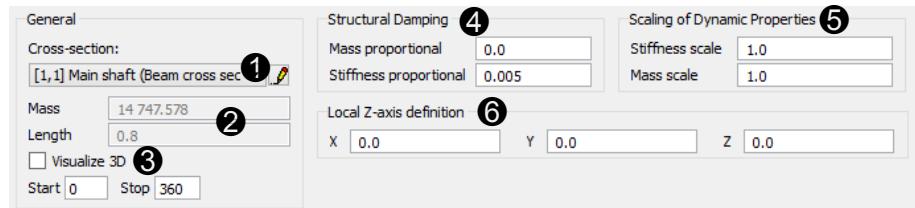
5

If you, e.g., type the value 10 and press **OK** (or hit the **Enter** key), then 9 new beam elements are created, inheriting all properties of the selected beam, which then will become the first of the 10 smaller elements that replace the selected element. This can be used to refine the discretization of a beam string when that is needed to obtain proper solution accuracy, for instance, due to complicated load patterns from hydrodynamic loads.

The **Split Beam...** command may also be applied repeatedly on the same beam string to obtain graded refinements in a localized area of the beam.

### 5.3.3 Beam properties

When a beam is selected in the Model Manager panel, its properties are displayed in the Property Editor panel (shown below).



- ① **Cross-section** – This pull-down menu enables the selection of the cross section object to be associated with this beam element, see [Section 5.4, "Beam cross sections"](#) below.
- ② **Mass and Length** – Displays the length and the total mass of the beam element. The length is computed from the coordinates of the end triads of the beam. The mass is computed from the beam length, and the cross section area and mass density that is assigned to the beam via the cross section object. These fields are updated automatically if any of the referred property objects are changed.
- ③ **Visualize 3D** – This toggle enables a 3D visualization of the beam, using the cross section parameters to render its actual geometry. The default visualization is a straight line connecting the two triads, with coloring depending on the referred cross section object. The *Start* and *Stop* fields are used to specify a sector of partial 3D visualization. They define the start and stop angles (in degrees), relative to the local Y-axis of the cross section, of the sector that is to be rendered in 3D.
- ④ **Structural damping** – Allows you to change the values of both the mass and stiffness proportional damping for the beam. These fields are equivalent to similar fields in the *Part tab* of the Part Property Editor panel, see [Section 5.1.5, "Part properties"](#).
- ⑤ **Scaling of dynamic properties** – Allows you to scale stiffness and mass of each individual part in the dynamics simulation. These fields are equivalent to similar fields in the *Part tab* of the Part Property Editor panel, see [Section 5.1.5, "Part properties"](#).
- ⑥ **Local Z-axis definition** – These three fields allow you to define a vector defining the local Z-axis of the local coordinate system of the beam. This vector is defined in the coordinate system of the parent sub-assembly of the beam, or in the global coordinate system in case of no parent assembly. The local Y-axis is then defined as the cross

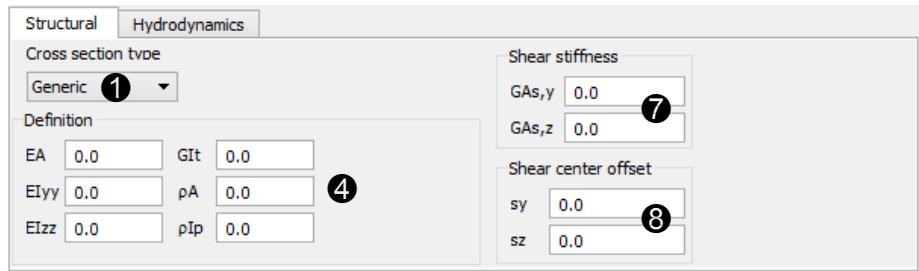
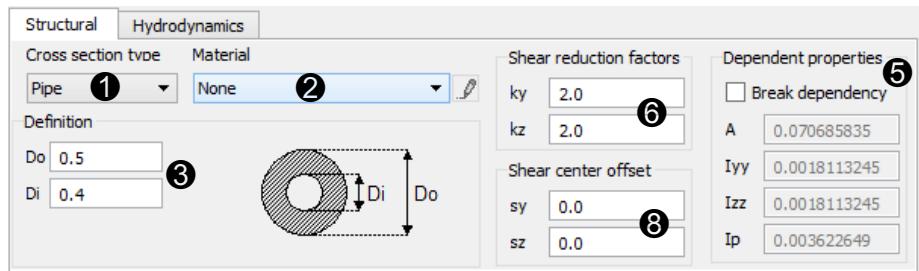
product between the Z-axis and the local X-axis, the latter being the unit vector from end 1 towards end 2 of the beam element.



**NOTE:** If the specified vector is parallel to the local X-axis, or if 0.0,0.0,0.0 is specified, a globalized coordinate system is used instead in which the local Z-axis is chosen to be as close as possible to the global Z-axis.

## 5.4 Beam cross sections

Beam cross sections are used to describe the cross section properties of a beam element, see [Section 5.3.3, "Beam properties"](#). When a Beam cross section is selected in the Object browser, its properties are displayed in the Property Editor panel, as shown below.



- ① *Cross section type* - This pull-down menu enables the selection of the cross section type. Currently you can select *Pipe* or *Generic*.
- ② *Material* - This pull-down menu enables the selection of the material property object that this cross section is to be associated with, see [Section 5.4.1, "Material"](#) below. The material selection is not available for the *Generic* cross section type, as the material properties then are embedded in the other cross section parameters.

- ③ **Definition** - Depending on the cross section type, you will have various fields for defining the cross section geometry. For *Pipe* sections, you specify the outer (*Do*) and inner (*Di*) diameters only.
- ④ For *Generic* cross sections, you specify the axial stiffness (*EA*), the bending stiffnesses (*Elyy* and *Elzz*), the torsional stiffness (*Gt*), the mass per unit length (*ρL*) and the torsional moment of inertia (*plp*).
- ⑤ **Dependent properties** - These properties (the cross section area, *A*, and the moments of inertias, *Ip*, *Iyy* and *Izz*) are updated automatically based on the specified geometry parameters. However, if the *Break dependence* toggle is enabled, any of the dependent properties may be overridden manually.
- ⑥ **Shear reduction factors** - These values specify the ratio between the effective transverse shear area and the actual cross section area, in the local *y*- and *z*-axis directions, respectively. Refer to text books on structural mechanics on how these parameters depend on different cross section shapes. If these values are set to zero, the transverse shear deformation is ignored in the FE model (infinite shear stiffness).
- ⑦ **Shear stiffness** - For *Generic* cross section, you specify the shear stiffness (*GAs,y* and *GAs,z*) directly, instead of the shear reduction factors. Non-positive values in these fields will be interpreted as infinite shear stiffness (no shear deformation).
- ⑧ **Shear center offset** - These fields allow you to specify the local position of the shear centre of the cross section with respect to its elastic centre (or the neutral axis). The shear centre defines the point through which a transverse load must be act in order to avoid torsional deformation in the beam element. For symmetric cross sections the shear centre coincides with the elastic centre and the values 0.0 are appropriate.

#### 5.4.1 Material

Materials are used to describe material properties of beam elements (linear isotropic material law only). They are referred to via the Beam cross section objects described above. When a Material is selected in the Object browser, its properties are displayed in the Property Editor panel, as shown below.

p	7 850.0
E	2.1e+11
v	0.29
G	8.1e+10

- $\rho$  - Mass density
- $E$  - Young's modulus of elasticity
- $\nu$  - Poisson's ratio
- $G$  - Shear modulus of elasticity



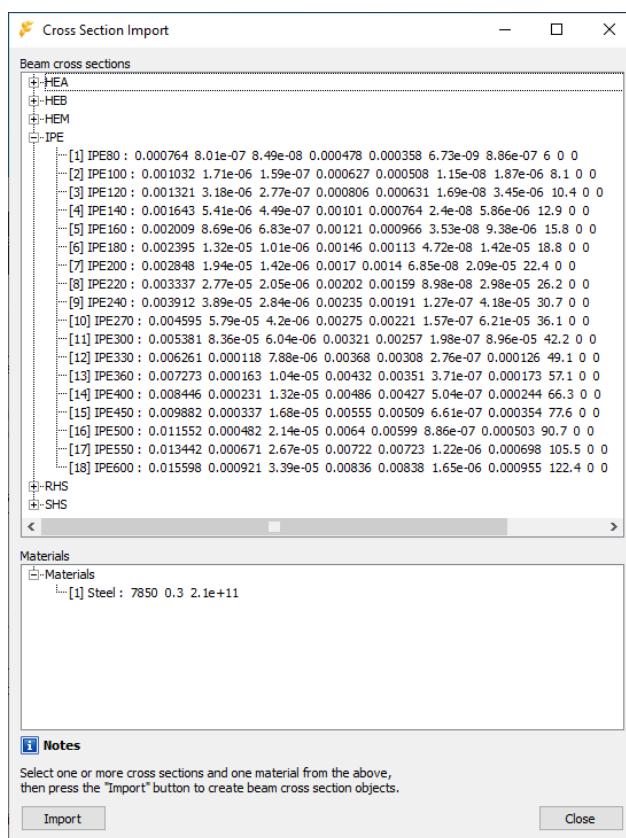
**NOTE:** Only one of the parameters  $\nu$  and  $G$  needs to be specified, as they are connected through the relation  $G = 0.5E/(1+\nu)$ . Whenever you change one of these values, the other value will update automatically.

## 5.4.2 Import of generic cross sections

The Fedem installation is equipped with a database of predefined beam cross sections and materials, which can be used to quickly load the cross section properties of your need into the model.



To access this database, select **Import Beam Cross Sections...** from the *File* menu. The Cross Section Import dialog box (shown below) pops up.



You can here select one (or more, hold down the **Shift** or **Ctrl** key to multi-select) items from the *Beam cross sections* list, together with one item from the *Materials* list, and press the **Import** button to generate generic beam cross section objects in your model with those properties.

The cross section database is stored as a set of *.csv* files (tab-separated columns) in the “Properties/CrossSections” sub-folder in your Fedem installation. Each line in these files correspond to one item in the Cross Section Import dialog box, and there is typically one file for each group of cross section types. You may easily expand the cross section database by creating files with similar format placed in a “CrossSections” sub-folder in the home directory of your user profile. They will then be appended to the pre-installed list of cross sections when you open this dialog box.

## 5.5 Triads



To construct a working mechanism, parts and/or beams are connected to each other using elements such as joints, springs and dampers. Fedem uses a modeling object called a *triad* to make these connections. Triads are also used as attack point for external point loads on parts, and concentrated additional masses. The triad defines a set of three mutually perpendicular coordinate axes originating from the connection point.

Triads enable parts to be connected to other mechanism elements using the parts' FE nodes as connection points. When parts are joined together in your model, the connection points are retained after the model reduction process as external FE nodes (see the Fedem 7.6 Theory Guide, *Chapter 3, "Model Reduction"*). During simulation, a triad must move rigidly in translation and rotation with the FE node to which it is attached. This means that triads (and therefore connections) can only be placed on FE nodes with six DOFs, since three-DOF solid nodes allow random rotation.

### 5.5.1 Triads in joints

Triads are used to connect joints to parts and beams in the model in the same way that a door hinge uses one hinge-plate to attach the hinge to the door and the other to attach the hinge to the frame. Each type of joint may use a different number of triads to make the required connections. When a joint is created, its triads are created along, and positioned automatically. (See [Section 5.6, "Joints"](#) for more information about how triads are used in joints.)

## 5.5.2 Triad symbols

Because triads can be used for several different purposes—as building blocks for joints; attachment points for springs, dampers, and loads; and measuring points for sensors—different symbols are used to visualize the triads based on its usage.



**TIP:** When attached, all triads are the same color, regardless of the symbol used to represent them. To identify a triad, simply look for the triad color (default = green). (Click the **General Appearance** button to access options for changing the color of triads and other mechanism elements.)

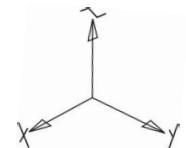
### Diamond

A triad is depicted as a small diamond (shown at right) when the triad's *coordinate system* is *not* used in your mechanism—for example, when a triad is placed on an FE node and no other mechanism element is attached, or when it is only used to attach a load, torque, axial spring, or damper.



### Coordinate system

A triad is visualized as a coordinate system (shown at right) when the triad stands alone, and its coordinate system is referenced in one of the following ways:



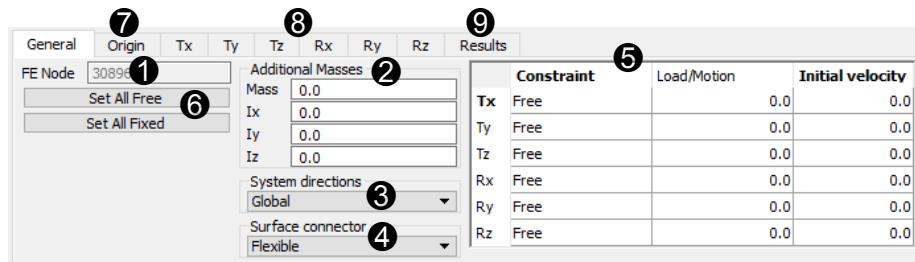
- The triad's coordinate system is used by a sensor to measure a variable with components defined in the local (triad) coordinate system (see [Section 5.12, "Sensors"](#)).
- The triad's coordinate system is used to define mass or inertia components for the triad (see [Section 5.5.3, "Triad properties"](#) below).
- The triad's coordinate system is used to define boundary conditions for use in the initial equilibrium analysis and/or the system eigenvalue analysis (see [Section 5.5.3, "Triad properties"](#) below).

### Member of a Joint

The triads that are members of joints are visualized as integral parts of the different joint symbols. Refer to [Section 5.6, "Joints"](#) for details.

### 5.5.3 Triad properties

To view the properties of a triad, select it in the *Modeler* view or Model Manager *Objects* list. The Property Editor panel for a triad consists of six tabs for each of the triad DOFs where their properties are displayed in detail, in addition to the *General* tab (shown below), the *Origin* tab and the *Results* tab.



- ①** *FE Node* – This field displays the ID number of the FE node to which the selected triad is attached.

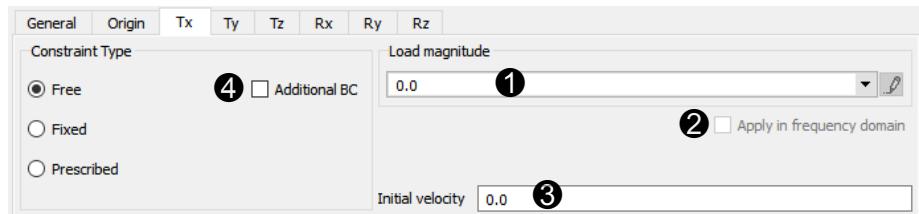


**TIP:** The FE node number can be useful if you edit the part (.fcl) file manually (described in [Appendix A, "FE Model Interface"](#)).

- ②** *Additional Masses* – These fields enable you to apply additional mass and inertia to the triad.
- ③** *System directions* – This pull-down menu enables selecting different reference coordinate systems for the DOFs associated with the triad.
- *Global* - The global inertial coordinate axes are used.
  - *Local, initial* - The local axis directions of this triad in its modeling configuration is used.
  - *Local, withrotated* - The local axis direction of this triad that follows the deformation of the model during simulation is used.
- ④** *Surface connector* - If the triad has a surface connector associated with it (see [Section 4.6.2, "Surface Connectors"](#)), you may here switch the connector type between *Flexible* and *Rigid*.
- ⑤** *Summary table* – An overview of all the DOF properties in the triad is provided here. This table is for viewing only (can not be edited).
- *Constraint* - Shows the *Constraint* type selected for this triad DOF.
  - *Load/Motion* - Shows the applied load or prescribed motion.
  - *Initial velocity* - Shows the initial velocity at this triad DOF.

- ⑥ Set All Free/Fixed - These two buttons (not present for slave triads) enable you to quickly set the Constraint type of all triad DOFs to either *Free* or *Fixed*, without the need to go into each of the DOF tabs.
- ⑦ Origin – This tab contains the *Origin properties* of the triad. See [Section 4.5.4, "Origin property"](#).
- ⑧ Triad DOF tabs – This tabs display the detailed properties related to each of the six DOFs associated with the triad. The available options here depends on the selected *Constraint type*, as shown below.
- ⑨ Results – This tab contains toggles for activating output of certain solution variables related to the triad, see "[Results tab](#)" below.

### Free DOF



- ① *Load magnitude* – You can apply a load in this triad DOF. This will be a torque or a force depending on whether this is a translational or a rotational DOF.
- ② *Apply in frequency domain* – If a *Function* is specified in the *Load magnitude* field, this toggle will enable the load to be applied in the frequency domain instead of the time domain, if the *Frequency response analysis* toggle is enabled in the Dynamics Solver Setup dialog box, see [Section 8.5.1, "Dynamics Solver \(Basic Mode\)"](#).
- ③ *Initial velocity* – You may specify a velocity that should be applied as initial condition in this DOF, i.e., the actual velocity at the start time of the dynamics simulation.
- ④ *Additional BC* – When this toggle is enabled, the movement in this triad DOF is fixed during the initial equilibrium analysis and optionally also in the eigenmode analysis. (See also "[Eigenmode tab](#)" in [Section 8.5.2](#) and the Fedem 7.6 Theory Guide, Section 7.8 "[Quasistatic equilibrium](#)" and Section 9.6, "[Eigenvalue results](#)".)



**TIP:** Sometimes during the creation of a complex model, it is useful to test the dynamics when all initial conditions are off (equal to zero). To facilitate this without having to manually remove all the defined initial conditions, the dynamic solver option `-ignoreIC` may instead be specified in the Additional Solver Options dialog box (in the Dynamics Solver field, see [Section 8.2.3, "Additional solver options"](#)). Then all defined initial conditions will be ignored (assumed zero).

### Fixed DOF

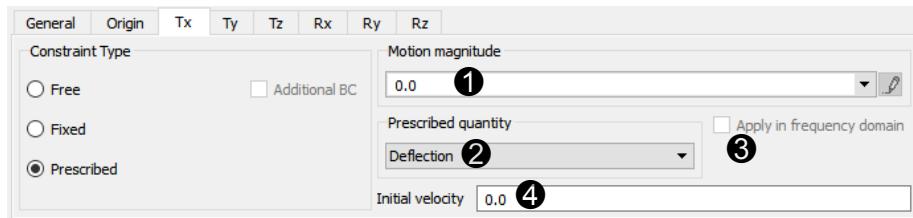


For Fixed triad DOFs, no further property settings are available.



**TIP:** You may plot the reaction force associated with a fixed or prescribed triad DOF by selecting the "Force" or "Moment" item under the Triad node in question from the RDB selector (see "[Selecting RDB results](#)" in [Section 9.2.4](#)).

### Prescribed DOF

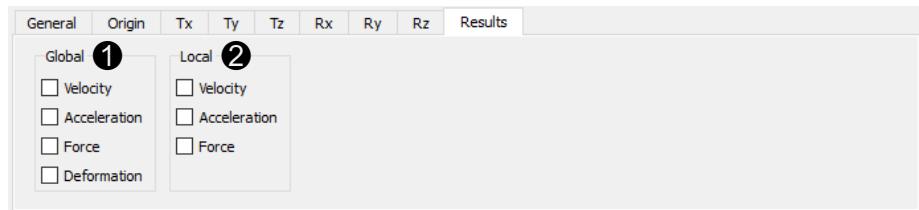


- ① *Motion magnitude* – You can prescribe a constant deflection, velocity or acceleration by typing the desired value into this field, or a time-dependent evolution of the motion by selecting a function.
- ② *Prescribed quantity* - You may choose whether you want to prescribe the *Deflection*, the *Velocity* or the *Acceleration* for the triad DOF.
- ③ *Apply in frequency domain* – If a *Function* is specified in the *Motion magnitude* field, this toggle will enable the prescribed motion to be applied in the frequency domain instead of the time domain, if the *Frequency response analysis* toggle is enabled in the Dynamics Solver Setup dialog box, see [Section 8.5.1, "Dynamics Solver \(Basic Mode\)"](#).
- ④ *Initial velocity* - You may specify a velocity that should be applied as initial condition in this triad DOF, i.e., the actual velocity at the start time of the dynamics simulation.



**CAUTION:** If using a prescribed non-constant motion in a triad DOF, it is the analysts responsibility to also specify an initial velocity that complies with the motion. Otherwise, fictitious transients may occur in the beginning of the simulation.

### Results tab



The *Results* tab of the Property Editor panel for a Triad (shown above) contains toggles for activating output of secondary solution variables to the results database for the triad, such that they are available for curve plotting, etc.

- ① *Global* - Activates output of quantities measured in the global coordinate system. *Deformation* represents the displacement (and rotation) of the triad, relative to the rigid body motion of the Part (or Beam) the triad is connected to. For triads that are not connected to a Part or a Beam, it will be equal to the total displacement of the triad.
- ② *Local* - Activates output of quantities measured in the local with-rotated coordinate system of the triad.



**Tip:** You don't need to enable these toggles if the solver option *-allSecondaryVars* or *-allTriadVars* is used. Then these quantities will be stored for all triads in the model.

5

## 5.6 Joints

As with real mechanisms, you may connect one part or beam to another using *joints*. A joint introduces motion and/or spring constraints between the two parts it is acting between. These constraints are applied on the *joint DOFs*, also called *Joint Variables*. Each Fedem joint uses at least two triads to connect the joint to parts. One or more of the joint's triads are labeled "master" while one triad is labeled "slave", with the constrained DOFs of the slave triad following the movement of the master(s). (See the Fedem 7.6 Theory Guide, *Chapter 6, "Modeling of Joints."*)

To attach a joint to parts, the joint's master triad is attached to an FE node on one part and the slave triad to an FE node on another part. This means that when the mechanism moves, the FE node (and part) on the slave side of the joint follows the motion of those on the master side. FE nodes and parts can, therefore, also be referred to as masters and slaves. See [Section 4.6, "Attaching and detaching elements"](#) and [Section 4.6.1, "Using the Attach command"](#) about how to attach joints.



**TIP:** To determine which triad is the master and which is the slave, select one of the parts or the joint to examine the Topology view of master and slave triads connected to the part/joint. You can then select (click and hold down the mouse button) the master/slave triad to highlight it in the Modeler view.

## 5.6.1 Joint variables

The joint variables are the accessible or controllable DOFs for each joint. As an example, the Revolute Joint normally has one accessible DOF, namely the rotation about one axis. The other DOFs are fixed. For most joints the DOFs that are not accessible are fixed, but for Prismatic and Cylindrical joints that is not the case. Refer to [Section 5.6.4, "Prismatic joint"](#) and [Section 5.6.4, "Cylindric joint"](#) for further details.

The behavior of the joint variable can be controlled or customized in several ways. There are four main options.

- Fixed - This DOF is fixed, and can not be moved. It is removed from the system of equations (condensed out).
- Free - This DOF is free to move. No constraints are applied.
- Prescribed - This DOF can be assigned a prescribed motion, and is thus condensed out from the system of equations.
- Spring-Damper - This DOF is free to move, but a spring and a damper may be applied to assign stiffness and/or damping properties to it.

### Integrated springs and dampers

When setting a joint variable to be spring and damper controlled, the joint springs and joint dampers are initially inactive (their properties—including spring stiffness and damper coefficient—are initially set to zero). You can then assign values to the joint's spring and damper properties in the Property Editor panel (see [Section 5.9, "Springs and Dampers"](#) for more information about the spring and damper behavior.)

The integrated springs and dampers are listed in the *Topology* view of the joint as separate items; they are however not listed in the *Objects* list of the Model Manager. You can access the full Property Editor panel for the joint springs and dampers by double-clicking on the entry in the *Topology* view, but the normal way of editing their properties is through the DOF tabs in the Joint Property Editor panel (see "[Joint variable properties](#)" below).

## 5.6.2 Joint properties

You can select a joint to display its properties in the *Property Editor* panel (shown below for a Free joint). For all joint types, the panel consists of one tab with a summary table of the major joint properties, and additional tabs for each of the *joint variables* where their properties are displayed in detail. The summary table shows a non-editable summary of all joint variables along with other editable joint properties. There is an *Origin* tab for the point-to-point joint types (Rigid, Revolute, Ball and Free joints) and an *Advanced* tab with further properties for Ball and Free joints. Finally, all joint types (except for Rigid joints) have a *Results* tab for output specification.

Since the number of joint variables depends on the joint type, you may see from zero (Rigid joint) to six (Free joint) sets of joint variables listed in the summary table and a similar number of joint variable tabs.

Constraint	Load	Model length	Init disp.	Length change	Init vel.	Sori	Sori scale	Damper	Dmd. scale
Tx Spr/Dmp	0.0	0.0	0.0		0.0	1.0			3.0
Ty Free	0.0	0.0			0.0				
Tz Free	0.0	0.0			0.0				
Rx Free	0.0	0.0			0.0				
Ry Free	0.0	0.0			0.0				
Rz Free	0.0	0.0			0.0				

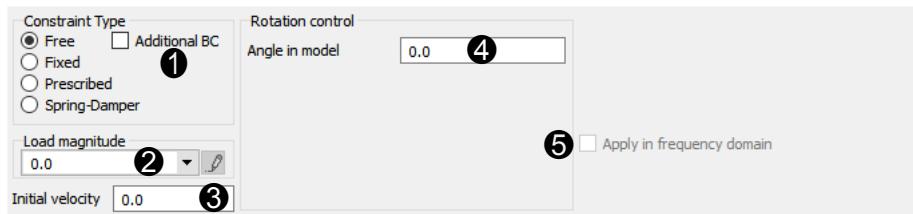
- ① *Summary tab* – This tab displays a *Summary table* over all properties of the joint.
- ② *Origin tab* - This tab contains the definition of the joint coordinate system, i.e., the position and orientation of its origin (see [Section 4.5.4, "Origin property"](#)). The joint variables are defined in this coordinate system. See also "[Moving point-to-point joints](#)" in [Section 5.6.3](#).
- ③ *Joint variable tabs* – These tabs display the properties related to the joint variable in question (see "[Joint variable properties](#)" below).
- ④ *Advanced tab* – This tab displays properties related to rotation formulation and spring inter-connectivity of the joint (see "[Advanced joint properties](#)" below).
- ⑤ *Results tab* – This tab contains toggles for activating output of certain solution variables related to the joint, see "[Results tab](#)" below.

- ⑥ *Friction* – Some joint types allow you to add friction properties to the joint by selecting from the list of frictions in your model (see [Section 5.8, "Frictions"](#)).
- ⑦ *Set All Free/Fixed* - These two buttons enables you to quickly set the Constraint type of all the joint DOFs to either *Free* or *Fixed*, without the need to go into each of the joint DOF tabs.

### Joint variable properties

The joint variable tabs display the different options and settings for each joint variable. The displayed options depend on the *Constraint Type*.

#### Free



- ① *Additional BC* - When this toggle is enabled, the movement in this joint variable is fixed during the initial equilibrium analysis and optionally also in the eigenmode analysis. (See also "[Eigenmode tab](#)" in [Section 8.5.2](#) and the Fedem 7.6 Theory Guide, [Section 7.8 "Quasistatic equilibrium"](#) and [Section 9.6, "Eigenvalue results"](#).)
- ② *Load magnitude* - You can apply a *Load* on the joint variable. This will be a torque or a force depending on whether the joint variable is a translational or a rotational DOF. The actual force value will be saved as a results quantity and thus available for plotting in a graph.
- ③ *Initial velocity* - You may specify a velocity that should be applied as initial condition in this joint variable, i.e., the actual velocity at the start time of the dynamics simulation.



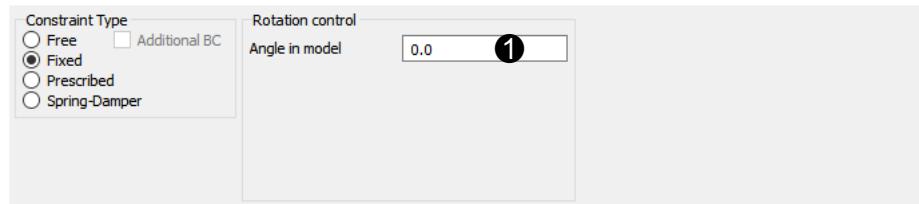
**Note:** The initial velocities of the slave triad in a joint are derived automatically in Fedem from the initial velocities of the joint DOFs and the master triad(s), based on the constraint equations representing the joint. However, it is also possible to specify the slave triad velocities explicitly and leave the joint DOF velocities zero. Fedem will then derive the initial joint DOF velocities by inverting the constraint equations of the joint. If non-zero initial velocities are specified in both the joint DOFs and slave triad DOFs, only the joint DOF values are used.

- ④ *Length/Angle in model* - This field shows the initial value of this joint variable as modeled. The value defines the initial configuration of this joint variable in the dynamics simulation. For rotational DOFs the

value can be edited to set a different initial rotation. The 3D view will then update instantly, showing the new rotation in the joint symbol.

- ⑤ **Apply in frequency domain** – If a *Function* is specified in the *Load magnitude* field, this toggle will enable the load to be applied in the frequency domain instead of the time domain, if the *Frequency response analysis* toggle is enabled in the Dynamics Solver Setup dialog box, see [Section 8.5.1, "Dynamics Solver \(Basic Mode\)"](#).

#### Fixed



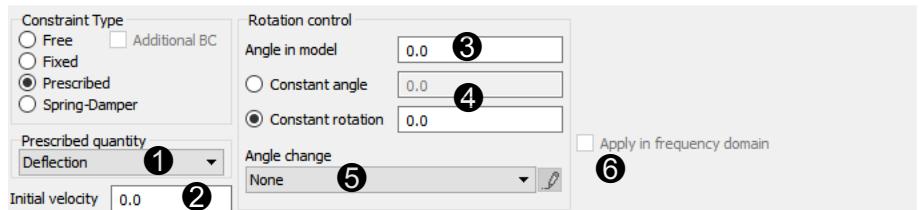
- ① **Length/Angle in model** - This field shows the fixed value of this joint variable as modeled. For rotational DOFs the value can be edited to set a different fixed rotation state. The 3D view will then update instantly, showing the new rotation in the joint symbol.



**TIP:** You may plot the reaction force associated with a fixed joint DOF by selecting the "Force/Moment value" item under the joint variable node in question from the RDB selector (see "Selecting RDB results" in [Section 9.2.4](#)).

5

#### Prescribed



- ① **Prescribed quantity** - You may choose whether you want to prescribe the *Deflection*, the *Velocity* or the *Acceleration* for the joint DOF.
- ② **Initial velocity** - You may specify a velocity that should be applied as initial condition in this joint variable, i.e., the velocity at the start time of the dynamics simulation.
- ③ **Length/Angle in model** - This field shows the initial value of this joint variable as modeled, and has the same interpretation as when the joint variable is *Free* (see above).
- ④ **Angle in model**
- ⑤ **Angle change**
- ⑥ **Apply in frequency domain**

- ④ Constant length/angle, Constant displacement/rotation** - These radio buttons and fields work together to allow you to set a constant prescribed value of the joint variable during the dynamics simulation. You can choose to enter the constant length/angle either as an absolute value, or relative to the *Length/angle in model*, if the *Prescribed quantity* is set to *Deflection*. For prescribed *Velocity* or *Acceleration*, you enter the constant velocity/acceleration that should be applied in the first of these two fields (the other field is inactive).



**CAUTION:** If you prescribe a length/angle that differs from the Length/Angle in model, this difference will be accounted for in the very first iteration of the dynamics simulation and thus lead to a dynamic shock effect. However, when the initial *Static equilibrium analysis* is switched on, the force due to this difference is taken as a pure static load and the transient shock should be avoided.

- ⑤ Length/Angle change** - You can prescribe the evolution of the motion during the simulation through a function. The function controls the change of the variable relative to the *Constant length/angle*, or the change of the velocity/acceleration relative to the *Constant velocity/acceleration*, in case *Velocity* or *Acceleration* is prescribed.
- ⑥ Apply in frequency domain** – If a *Function* is specified in the *Length/Angle change* field, this toggle will enable the prescribed motion to be applied in the frequency domain instead of the time domain, if the *Frequency response analysis* toggle is enabled in the Dynamics Solver Setup dialog box, see [Section 8.5.1, "Dynamics Solver \(Basic Mode\)"](#).



**TIP:** You may plot the reaction force and input energy associated with a prescribed joint DOF by selecting the "Force/Moment value" and "Input Energy" items under the joint variable node in question from the RDB selector (see "[Selecting RDB results](#)" in [Section 9.2.4](#)).

### Spring-Damper

Constraint Type <input type="radio"/> Free <input type="checkbox"/> Additional BC <input type="radio"/> Fixed <input type="radio"/> Prescribed <input checked="" type="radio"/> Spring-Damper  Load magnitude <input type="text" value="0.0"/> ②  Initial velocity <input type="text" value="0.0"/> ③	Stress free angle control ④ Angle in model <input type="text" value="0.0"/> <input checked="" type="radio"/> Constant stress free angle <input type="text" value="0.0"/> <input type="radio"/> Constant deflection <input type="text" value="0.0"/>  Stress free angle change <input type="text" value="None"/>	Spring properties ⑤ Stiffness <input type="text" value="0.0"/> <input type="button" value="..."/> Scale <input type="text" value="None"/> <input type="button" value="..."/>  Damper properties ⑥ Damping coefficient <input type="text" value="0.0"/> <input type="button" value="..."/> Scale <input type="text" value="None"/> <input type="button" value="..."/> <input type="checkbox"/> Use deformational velocity
---	--	---

These options enable you to add elastic and damping behavior to the joint variable by entering values for the spring and damper properties.

- 1 Additional BC** - When this toggle is enabled, the movement in this joint variable is fixed during the initial equilibrium analysis and optionally also in the eigenmode analysis. It has the same interpretation as when the joint variable is *Free* (see above).
- 2 Load magnitude** - You can apply a Load in the joint variable, similarly as when the joint variable is *Free* (see above).
- 3 Initial velocity** - You may specify a velocity that should be applied as initial condition in this joint variable, similarly as when the joint variable is *Free* (see above).
- 4 Stress free length/angle control** – This group of options concerns spring deflection calculation similar to the *Length/Angle control* of a *Prescribed* joint variable. See also [Section 5.9.1, "Spring properties"](#).
- 5 Spring properties** – This group of options concerns the spring characteristics, namely the relation between deflection and force. See [Section 5.9.1, "Spring properties"](#) for details.
- 6 Damper force/coefficient** – This group of options concerns the damper characteristics, namely the relation between velocity and force. See [Section 5.9.2, "Damper properties"](#) for details.

### Summary table

The summary table displays an overview of the settings for each joint variable. The columns relate to the fields in the [Joint variable properties](#) described above.

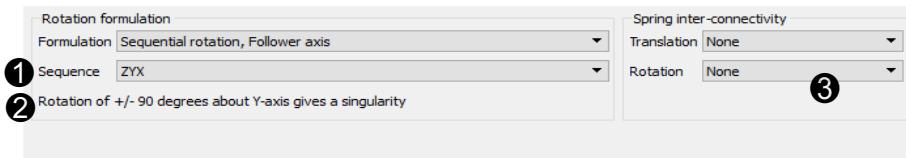
Summary	Origin	Tx	Ty	Tz	Rx	Ry	Rz	Advanced	Results
<b>Constraint</b>	Load								
<b>Tx</b> Spr/Dmp	0.0	0.0	0.0		0.0	1.0			3.0
<b>Ty</b> Free	0.0	0.0			0.0				
<b>Tz</b> Free	0.0	0.0			0.0				
<b>Rx</b> Free	0.0	0.0			0.0				
<b>Ry</b> Free	0.0	0.0			0.0				
<b>Rz</b> Free	0.0	0.0			0.0				

- Constraint** – Shows the *Constraint type* selected for the joint variable.
- Load** – Shows the load applied to the joint variable.
- Model length** – Shows the modeled length/angle of the joint variable.
- Init disp.** – Shows the initial deflection set up for the joint variable.
- Length change** – Displays the function, if any, that controls the change of length/angle of the joint variable. If the constraint type is set to *Prescribed* this change is directly applied to the joint DOF. If the Constraint type is set to *Spring-Damper*, however, this function controls the *Stress free length/angle change* of the spring.

- **Init vel.** – Shows the initial velocity applied to the joint variable.
- **Spring** – Displays the spring characteristics of the joint variable. Either a number describing a constant stiffness of the spring, or a description of the spring characteristics used as a non-linear stiffness- or force-deflection relationship.
- **Spr. scale** – Displays the function that is used to scale the force developed in the spring. Empty if no scaling is done.
- **Damper** – Displays the damping characteristics of the joint variable. Either a number describing a constant damping coefficient, or a description of a function used as a non-linear coefficient- or force-velocity relationship.
- **Dmp. scale** – Displays the function that is used to scale the force developed in the damper. Empty if no scaling is specified.

### Advanced joint properties

For the Ball joint and Free joint, you have possibility to alter the numerical formulation of how the rotational DOFs are represented internally. You can also control the *spring inter-connectivity*, a feature that can be used to describe the circular or cylindrical stiffness behavior of rubber bushings, etc. This is done through the *Advanced* tab shown below.



- ①** You may change the rotational formulation of the joint. The following choices are available:

- *Sequential rotation, Follower axis* - Euler angle parametrization.
- *Sequential rotation, Orthogonal axis* - Euler angle parametrization.
- *Rotational vector* - Singularity free Rodriguez parametrization.

See the Fedem 7.6 Theory Guide, *Section 2.3 "Finite rotations"* for further details on these choices.

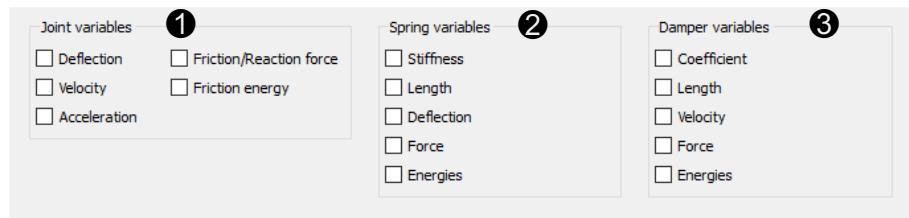
- ②** You may alter the update sequence of the Euler angle parameters. The default sequence is Z-Y-X.

**CAUTION:** The Sequential rotation formulation may lead to singularities in the rotation update computations if the joint undergoes a 90 degrees rotation about the local Y-axis. If you have such behavior in the joint, you must use the Rotational vector formulation to achieve proper results.



- ③ You may specify how the translational- and rotational joint springs should be inter-connected (Cylindrical or Spherical coordinates). This can be used to describe the cylindrical or spherical behavior of rubber bushings, or pin joints with clearances. If you want the spring characteristics in the joint variables to be interpolated resulting in a cylindrical/spherical behavior, select the proper setting from the drop-down menu. Please refer to the Fedem 7.6 Theory Guide, *Section 5.1.1 "Interconnected Spring Elements"* for details on how this affects the stiffness matrix.

### Results tab



The *Results tab* (shown above) contains toggles for activating output of secondary solution variables to the results database for the selected joint, such that they are available for curve plotting, etc.

- ① *Joint variables* - Activates output of the respective solution variables associated with each DOF in the joint.
- ② *Spring variables* - Activates output of the respective spring quantities in each spring/damper-constrained DOF in the joint. These toggles are not present if none of the joint DOFs are spring/damper-constrained.
- ③ *Damper variables* - Activates output of the respective damper quantities in each spring/damper-constrained DOF in the joint. These toggles are not present if none of the joint DOFs are spring/damper-constrained.



**TIP:** You don't need to enable these toggles if the solver option *-allSecondaryVars* or *-allJointVars* is used. Then these quantities will be stored for all joints in the model.

### 5.6.3 Point-to-point joints

With point-to-point joints, the motion constraints of the joint are applied between two points represented by the slave and the master triad with their corresponding FE nodes. Point-to-point joints are found on the *Mechanism Creation* tool bar (shown at right).

Each of the point-to-point joint types are described in the following.



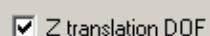
#### Revolute joint



The revolute joint has a single DOF that allows rotation of one part with respect to another about a common axis. Its joint variable is the angle from the master triad to the slave triad about the common z-axis (defined by the right-hand rule).

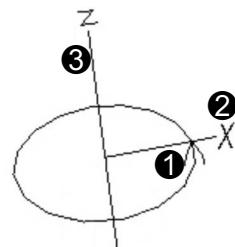
The Revolute joint has an optional Joint variable too; namely the translation along the common z-axis.

This Joint variable can be toggled on or off on the *Summary* tab of the Revolute joint Property Editor panel.



The symbol for a revolute joint is displayed in the *Modeler* view as shown below.

- ① The arrow represents the slave triad and indicates the positive direction for the joint angle.
- ② The straight line (labeled X) represents the master triad.
- ③ The revolute axis is the common z-axis of the master and slave triads. Together with the circle it represents the joint itself.

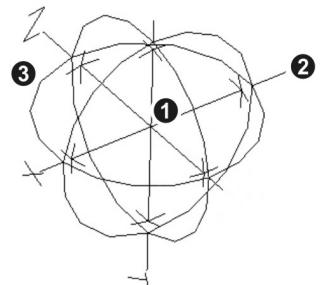


You can add friction to a revolute joint by selecting one from the list of frictions in your model in the *Friction* pull-down menu, located on the Property Editor panel. (See also [Section 5.8, "Frictions"](#) and the Fedem 7.6 Theory Guide, [Section 6.5 "Joint Friction"](#).)

### Ball joint



The ball joint has three DOFs that allow rotation of one part with respect to another about three axes. The joint variables are defined by the angles between the master triad and the slave triad in the x-, y-, and z-directions. The symbol for a ball joint is displayed in the *Modeler* view as shown to the right.



- ① The cross in the middle of the sphere represents the slave triad.
- ② The lines extending out of the sphere represent the master triad.
- ③ The circles represent the joint itself.

You can add friction to a ball joint by selecting one from the list of frictions in your model in the *Friction*



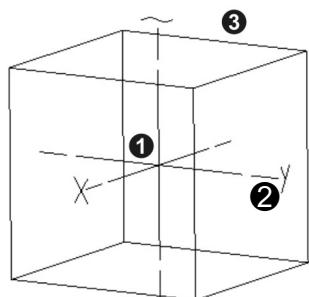
5

pull-down menu. Then you also need to select which one of the three joint DOFs that shall receive the friction moment. The effective normal load in the friction is then computed from the other two joint DOFs that are orthogonal to the selected DOF. (See also [Section 5.8, "Frictions"](#) and the Fedem 7.6 Theory Guide, [Section 6.5 "Joint Friction"](#).)

### Rigid joint



The rigid joint constrains all displacement between two parts, and is therefore used as a stiff connection. It has no joint variables. The symbol for a rigid joint is displayed in the *Modeler* view as shown to the right.



- ① The cross in the middle of the cube represents the slave triad.
- ② The lines extending out of the cube represent the master triad.
- ③ The cube represents the joint itself.

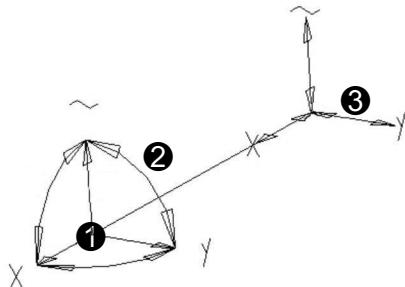
### Free joint



The free joint has six joint variables. The free joint can thus be used to introduce any type of mechanism motion constraint by setting the constraint type of each joint variable to fit your needs.

The symbol for a free joint is displayed in the *Modeler* view as shown to the right.

- ① The coordinate system in the lower left (straight arrows) represents the master triad.
- ② The rounded arrows together with the line between the two coordinate systems represent the joint itself.
- ③ The coordinate system in the upper right with double arrows represents the slave triad.



You can add friction to one of the free joint DOFs by selecting one from the list of frictions in your model in the *Friction* pull-down menu. The list of selectable frictions depends on whether you have selected a translational or a rotational dof in the *Joint DOF* pull-down menu.



The effective normal load in the friction is then computed from the two joint DOFs that are orthogonal to the selected DOF. (See also [Section 5.8, "Frictions"](#) and the Fedem 7.6 Theory Guide, [Section 6.5 "Joint Friction"](#).)

### Moving point-to-point joints

The point-to-point joint types have three parts that either can be moved independently, or as a whole. To turn on and off this behavior a group of options are available on the joints *Origin* tab. The two toggles (shown below) control whether the slave and/or the master triad will move along with the joint symbol if the joint itself is moved. (See also [Section 4.5.4, "Origin property"](#) and [Section 4.5.2, "Align CS and rotations"](#).)



The sensitivity of the *Position* and *Orientation* fields in the *Origin* tab of the joint and its triads will reflect the movability of the selected object, and may change when changing these options. E.g., triads attached to FE nodes can not be moved, and thus if the triad is set to follow the joint, the joint can not be moved either.



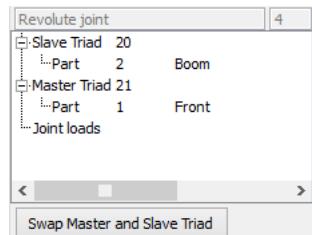
**NOTE:** These settings do not apply when you are using the **Smart Move** command to move the joint (see [Section 4.5.1, "Smart Move"](#)). When applicable, the **Smart Move** command will always move the master triad along with the joint.

The **Slave triad follows joint** toggle affects the *Position* of the slave triad only. The *Orientation* of the slave triad in a point-to-point joint will always follow that of the joint itself. The joint rotation variables are defined as the rotation between the joint coordinate system and the slave triad coordinate system and thus the triad rotation is controlled by the rotational joint variables alone. When creating a point-to-point joint, the default value of the rotational joint variables is zero.

### Swapping Master and Slave triads

After a point-to-point joint has been attached, there might be situations where it is desirable to swap the master and slave triads in the joint. This is needed, for instance if the slave triad of one joint also needs to be the slave in another joint. This is not possible in general. However, by swapping the master/slave definition of the first joint, such that its slave triad becomes the master instead, we can next use the same triad as a slave in another joint.

To swap master and slave triad in a joint, click the **Swap Master and Slave Triad** button located below the ID and Topology panel (shown at right). You will then notice that the ID's of the Slave and Master triads in the *Topology* view are swapped as well. It is not possible to swap triads in joints that are not fully attached, and (of course) not for joints where the master is attached to ground.



## 5.6.4 Point-to-path joints

Point-to-path joints are more complex than point-to-point joints as they require more than one master triad for each slave. The motion is defined by at least two master triads in a straight or curved path. Point-to-path joints are found on the *Mechanism Creation* tool bar (shown at right).



**NOTE:** The same master triads can be used in more than one point-to-path joint.

Each of the point-to-path joint types is described in the following.

### Prismatic joint



A flexible prismatic joint consists of a slave triad sliding along a straight path defined by two or more master triads. The local coordinate system of the joint is defined with its z-axis directed along the slide path. The x- and y-axes are defined from the coordinate systems of the master triads.

The joint has three unconstrained DOFs, but only a single joint variable (the slider variable) that allows you to control the translational displacement of the slave along the local z-axis. Rotation is constrained about the z-axis, but not in the other two directions (the slave can rotate about the local x- and y-axes independently of the masters).

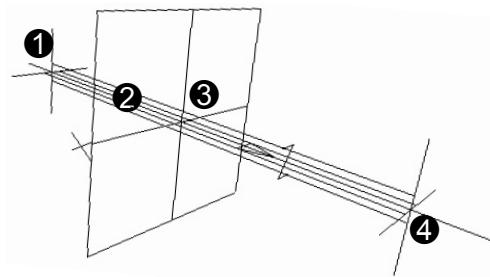


**TIP:** You can attach two prismatic joints to make a stiff translating connection by attaching the masters for the two joints to the same nodes (in the same order) and attaching the two slave triads to the same part on different nodes.

The joint variable for prismatic joints is the distance from the first master to the slave triad in the direction of the local z-axis.

The symbol for a prismatic joint is displayed in the *Modeler* view as shown to the right.

- ① First master triad
- ② The slider path (represented by a line from the first to the last master)
- ③ Slave triad
- ④ Last master triad



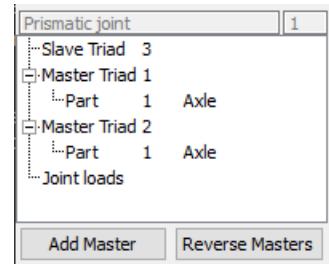
### Adding masters

A prismatic joint is created by selecting the position of the first and last master triad. The slider path is then defined as the straight line between the two triads. However, the slider path may be redefined by adding more master triads along that line. This improves load distribution during the simulation as the forces from the slave triad are distributed to the two masters closest to the current position of the slave.

To add master triads to a joint, click the **Add Master** button located below the ID and Topology panel (shown at right) and select additional FE nodes along the slider path.



**NOTE:** You can add master to a prismatic joint only after it has been attached to a part (see [Section 4.6.1, "Using the Attach command"](#)). It is not possible to add masters to a joint that is attached to ground.



### Reversing masters

The direction of a prismatic joint, i.e., the direction of its local Z-axis, can be swapped by using the **Reverse Master** button located below the ID and Topology panel (shown above). You can do this also for joints that are attached to ground, and also before it is attached.

### Adding friction

You can add friction to prismatic joints by selecting one from the list of frictions in your model in the *Friction* pull-down menu, located on the Property Editor panel. (See also [Section 5.8, "Frictions"](#) and the Fedem 7.6 Theory Guide, [Section 6.5 "Joint Friction"](#).)

### Cylindric joint

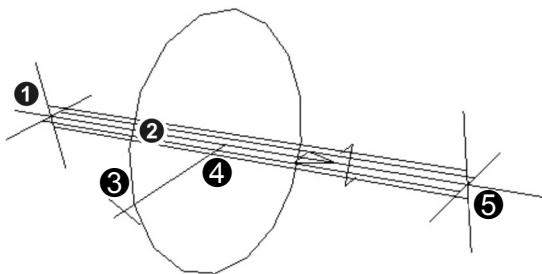


A flexible cylindric joint has four unconstrained DOFs that allow both translational displacement along the local z-axis, and rotation about the local z-axis. As with prismatic joints, cylindric joints do not constrain motion in the other two rotational directions. The joint's local coordinate system is defined in the same manner as for the prismatic joint.

The cylindric joint has two joint variables. They are the translational distance along the local z-axis from the first master to the slave (the slider variable) and the angle of rotation of the slave about the local z-axis. The rotation angle is measured between the x-axis of the first master triad and the x-axis of the slave triad.

The symbol for a flexible cylindric joint is displayed in the *Modeler* view as shown below.

- ① First master triad
- ② The slider path  
(represented by the line from the first master to the last)
- ③ Rotational joint variable  
(represented by the angle of the x-axis)
- ④ Slave triad
- ⑤ Last master triad



You can constrain the two joint variables of a cylindric joint in a screw-like connection by defining a ratio of translational to rotational motion, called the *screw ratio*. This ratio determines how fast the slave rotates as it translates along the joint. To constrain the translational and rotational DOFs of a cylindric joint, enable the *Screw Connection* option in the Property Editor panel and assign a value to the *Screw Ratio*. See the Fedem 7.6 Theory Guide, Section 6.4.3, "Screw joint" for more information about the screw ratio.

Screw connection	
<input type="checkbox"/> Output Ratio	0.0



**TIP:** You can refine the slider path by adding master triads and reversing masters in the same way as for prismatic joints (see "[Prismatic joint](#)" above).



**TIP:** A zero screw ratio makes the cylindric joint equivalent to a prismatic joint.



### Cam joint

A cam joint has six unconstrained DOFs that allow the slave triad (called the *follower*) to move over a curved surface (called the *cam surface*). The cam surface is defined by a curve consisting of three-point circular arcs. Each arc is defined by the location of three master triads, also called *cam triads*. A cam joint must consist of one slave/follower triad and at least three master/cam triads. See also the Fedem 7.6 Theory Guide, Section 6.3.3, "Cam joint".

It is recommended to use at least one arc segment per quarter of a circle to make the solution more stable. This means you will need at least 8 master triads for a complete circle.



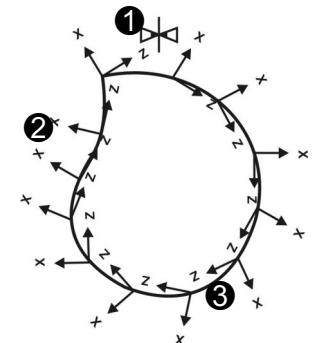
**TIP:** You can use the same cam triads in several different cam joints, making it possible to constrain several follower triads to the same cam surface.

An example cam joint is shown to the right.

- ① Slave/follower triad
- ② Master/cam triads (represented by the sets of x-, y-, and z-axes extending from the curve)
- ③ Cam curve (represented by the curve)

#### Creating cam joints

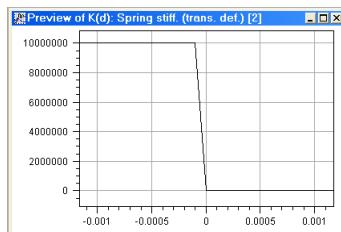
The cam curve is defined by circular arcs and straight lines. Each three-point arc is defined by three triads. If the triads are located on a straight line, a straight line will be defined (circular arc with zero curvature).



To create a cam joint, complete the following steps:

1. Click the cam joint icon.
2. Select a position for a new follower triad, or select an existing triad.
3. Confirm by pressing Done. If an existing triad was selected, this triad will become the follower triad, otherwise a new triad is created.
4. Select a position for the first master, or select an existing triad. You can also select an existing cam curve.
5. Confirm by pressing Done. If an existing triad was selected, this triad will become the first cam triad, otherwise a new triad is created. If an existing cam curve was selected, the new cam joint is complete, and the selected cam surface will be used by the new follower triad.
6. Repeat steps 4 and 5 until a sufficient number of masters has been added. As you add triads, they will be oriented automatically to have sensible orientations related to the cam curve. The z-direction is set to point along the cam curve, while the x-direction, considered to be "up", is calculated from the direction going from the master triad closest to the follower and to the follower triad.
7. To close the cam loop, add the first master triad as the last one.
8. Fedem tries to set sensible directions on the master triads created, but should any of the directions be inconvenient, rotate them using one of the tools to move mechanism elements. See [Section 4.5, "Moving mechanism elements"](#).

9. Define the spring characteristics you need for the contact behavior, and assign them to the correct joint variables. Normally, a non-linear spring with a stiffness-deflection curve as shown in the picture to the right will provide a decent contact behavior when assigned to the x-translation DOF.



#### **Local coordinate system**

The local coordinate system for a cam joint has its origin on the cam curve at a point calculated as the closest point to the follower; this point is referred to as the *contact point*. The local x-axis is then defined to be perpendicular to the cam surface and the z-axis tangential to the cam curve. The orientation of the local coordinate axes depends thus on the location of the contact point along the cam curve.

#### **Cam joint variables**

Cam joints display all the six DOFs as joint variables in the Property Editor panel, but have some restrictions on the *Constraint Type* setting that is unique for cam joints. The only legal settings are *Free* and *Spring-Damper*. The *Fixed* and *Prescribed* settings are not available because the cam joint uses a different formulation than the other joints.

The three main joint variables, defined in the x-, y- and z-directions of the cam joint's local coordinate system, are:

- X-position: The distance from the contact point to the follower in the direction normal to the cam surface (the "thickness" direction).



**TIP:** If no stiffness is assigned to the X-translation DOF, the whole cam joint will be completely ignored by the Dynamics Solver. This might be used as a simple tool to toggle a cam joint on and off during testing and modeling of complex models. The solver issues a warning when the X-translation spring is missing.

- Y-position: The distance from the contact point to the follower in the direction tangential to the cam surface and normal to the cam curve ("width" direction).
- Z-position: The distance along the cam curve from the first cam triad to the contact point (the slider variable).

You are also allowed to Spring-Damper constrain the rotational DOFs of the cam joint. Such rotational stiffness/damping might be beneficial as a stabilization tool in some cases.



**WARNING!** The rotational DOFs in a Cam joint are not suited for representing large rotations. However, this affects the solution only when some of these DOFs are Spring-Damper constrained. Therefore, when Spring-Damper constraining the rotational DOFs, you must ensure that the added stiffness is high enough to keep the rotations "small", typically  $Rx < 0.3$ ,  $Ry < 0.6$  and  $Rz < 3.0$  radians. If not, the solution will probably diverge.

The initial values of the cam joint variables are interpreted differently compared with the other joint types. The *Length/Angle in model* quantity is always zero for all variables, regardless of the modeling position of the follower. For the Tx and Ty DOFs, this means that the deflection is always calculated as the distance from the contact point to the follower in the local x and y directions, respectively. However, for the Tz, Rx, Ry and Rz DOFs, the deflection is measured relative to the modeling position of the follower. The stress free length/angle of any springs associated with these DOFs are then also defined relative to these initial positions.



**WARNING!** If the follower is not within the contact domain of the cam joint (see "[Cam thickness and width as contact domain](#)" below) at the beginning of the first time step, the rotational springs as well as the slider spring, if any, are ignored throughout the simulation. This happens because the stress free length of these springs then are undefined. A warning is issued from the dynamics solver if this occurs.

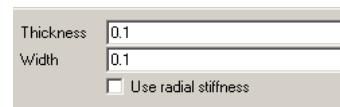
5

### Cam friction

The friction parameters for cam joints are the same as those for prismatic joints with the exception of the equivalent force, which is the sum of the x-spring and x-damper force in the cam joint. The friction state depends on the slider variable only.

### Cam thickness and width as contact domain

The thickness and width parameters shown in the Property Editor panel define a rectangular domain in the xy-plane of the local coordinate system and is used to determine whether it is necessary to test if the follower is in contact or not. The springs and dampers associated with joint variables are activated only when the follower is located within the distance *Thickness*/2 from the cam curve in the local x-direction and within the distance *Width*/2 in the local y-direction. Use of a "reasonable" thickness is of great importance to ensure that the contact springs are attached to the correct cam segment (a cam segment is the part of a cam curve between two triads). One should avoid having the cam thickness so large that two cam segments have overlapping contact domains.





**CAUTION:** When assigning highly non-linear spring characteristics to the cam joint variables to model contact behavior, it is often necessary to assign some associated damping to reduce fictitious oscillations due to sudden activation and deactivation of contact spring forces. A constant damping coefficient is then sufficient as long as the follower is within the contact domain throughout the simulation. However, if the follower enters the contact domain once or several times during the simulation, numerical instabilities may occur due to the sudden activation of the joint variable dampers, because they are active only when the follower is within the contact domain. To avoid this, it might be necessary to scale the damping coefficient with a function (see [Section 5.9.2, "Damper properties"](#)), that varies gradually from zero as the follower enters the contact domain, to one as the contact stiffness is activated.

### **Radial contact springs**

By enabling the *Use radial stiffness* toggle, the springs associated with the x- and y-variables are referred to local polar coordinates in the xy-plane instead. Thus, the x-coordinate is then the radial distance from the cam curve to the follower, and the y-coordinate is the angle between the local Cartesian x-axis and the axis extending from the contact point through the follower. The contact domain will consequently be a circular cylinder instead of a rectangular one, and the *Thickness* and *Width* parameters above will now define the radial and the angular (in degrees) extension of the contact domain. This can be used to simulate contact in pipes, etc.



**NOTE:** The *Use radial stiffness* toggle does not affect the dampers (if any) that are assigned to the joint variables. They are still applied in the local Cartesian coordinate system. It is therefore advisable to apply the same damping characteristics to the x- and y-variables when using radial stiffness, to ensure a proper damping behavior in the cam joint.

### **Cam with spherical or cylindrical follower**

Quite often the follower in a cam joint has some sort of spherical or cylindrical shape. This is not fully supported by Fedem, but this section describes how you can do it.

The radius of the sphere or cylinder must be entered as an *Initial stress free length* for the spring in the X-translation DOF (see [Section 5.9.1, "Spring properties"](#)). A normal contact stiffness function can then be used. The *Thickness* of the cam must also be set to a value greater than the roller radius in this case.

This will work as expected as long as the follower never is in contact with the cam curve at more than one location simultaneously. This means that the follower can not pass the inside of a v-shaped cam curve, or curve segments that have a radius equal to, or less than the roller radius. By

trying to do so, the numerical simulation will normally fail to converge when two simultaneous contact locations would be expected.

If the cam curve to be modeled has this kind of features, you will need to model the different parts of the contact curve as *separate cam joints* instead, and re-use the same triad as follower in all those cam joints. You will also have to set them up with the same contact spring characteristics and *Initial Stress free length*.



**CAUTION:** When using a radius on the follower, even **small** discontinuities of the cam tangent between curve segments might result in a v-shaped curve. The v's can cause numerical problems if the follower is on the inside of it.

## 5.7 Joint pair constraints

Joint pairs available in Fedem include Gear and Rack-and-Pinion objects. In addition, you can create general couplings using *Generic DB* objects.

5

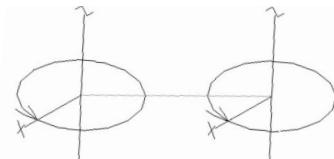
### 5.7.1 Gear



A gear is a rotational constraint between two revolute joints. The gear constrains the two joints to rotate at a given transmission ratio.

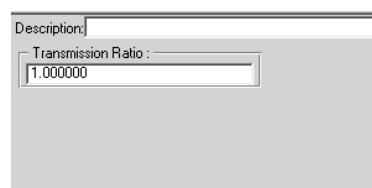
#### Gear symbol

The gear symbol (shown at right) is displayed in the *Modeler* view as a line between two revolute joints.



#### Transmission ratio

You can specify the gear transmission ratio (dimensionless rate) in the Property Editor panel (shown at right). For information about the gear transmission ratio, see the Fedem 7.6 Theory Guide, *Section 6.4.1, "Gear joint"*.



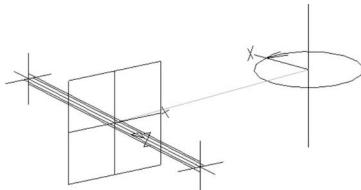
### 5.7.2 Rack-and-Pinion



A Rack-and-Pinion is a constraint between a prismatic and a revolute joint. The complete system is considered a five-DOF joint that constrains only a rotational input displacement to a translational output displacement. The joint has a transmission ratio similar to that of gears.

### Rack-and-Pinion symbol

The Rack-and-Pinion symbol (shown at right) is displayed in the *Modeler* view as a line between a prismatic joint and a revolute joint.



### Transmission ratio

You can specify the transmission ratio (dimensionless rate) for a Rack-and-Pinion constraint in the Property Editor panel (see [Section 5.7.1, "Gear"](#)).

### 5.7.3 General joint pair constraints

In addition to the specific cases [Gear](#) and [Rack-and-Pinion](#), any other joint pair constraint can be modeled using a Generic database object. See [Section 5.14, "Generic database objects"](#) on how to create and manipulate such objects.

A joint pair constraint is created by using the *Type* keyword "HIGHER\_PAIR" and by specifying the two joints and DOFs to be coupled in the *Definition* field, as in the example shown to the right. The transmission ratio between the two joint DOFs is specified using the "coeff" keyword in the *Definition* field, as shown.

Description	generell kobling
Type	HIGHER_PAIR
Definition	slaveJoint = 17 slaveJointDof = 3 masterJoint= 12 masterJointDof = 3 coeff = 0.5

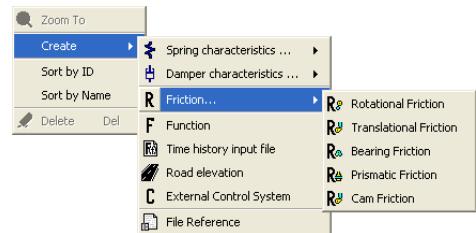


**NOTE:** The ID numbers used in the joint identification are their base ID. (You can find the base ID of an object using the Object Browser, see [Section 2.4.6, "Object Browser"](#).)

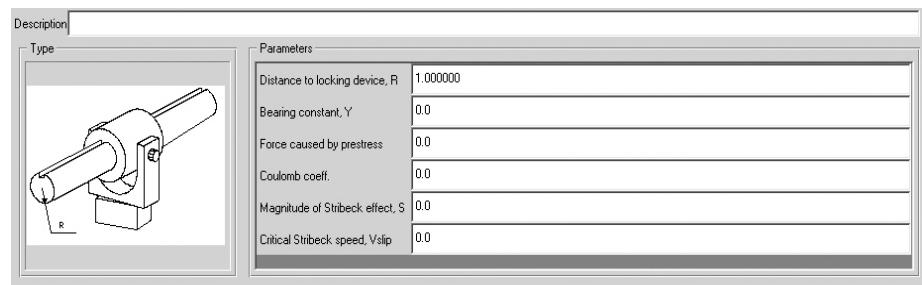
## 5.8 Frictions

Joint friction is based on the forces, moments, and velocity in a joint. These forces and moments give an equivalent load, which is the basis for computing the friction force. For a detailed description of friction behavior, see the Fedem 7.6 Theory Guide, *Section 6.5, "Joint friction"*.

You can create a friction by right-clicking an empty space in the Model Manager *Objects* list, selecting **Create, Friction** and then the desired friction type. You can also access this command from the *Mechanism* menu in the main window.



Frictions are managed in the Model Manager *Objects* list. If you have created frictions, you can expand the *Friction* group to see a list of the frictions in your model. Selecting a friction from the *Objects* list displays the friction properties in the Property Editor panel. Each type of friction has a different image and parameters associated with it. The figure below shows the Property Editor panel with prismatic joint friction selected.



To edit the friction, enter new values for each of the friction parameters listed in the Property Editor panel. See the Fedem 7.6 Theory Guide, *Section 6.6, "Joint friction"* for a description of friction parameters.



**TIP:** To associate the friction with the appropriate joint, select the joint and edit its friction properties in the Property Editor panel.

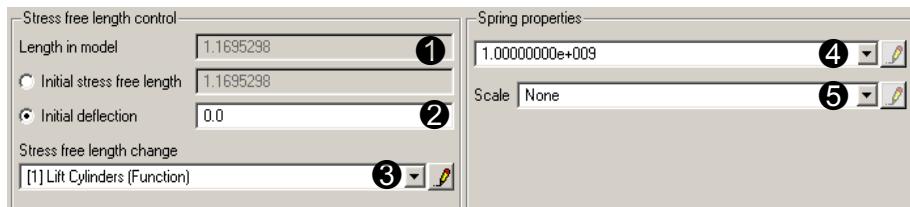
## 5.9 Springs and Dampers

There are two types of springs and dampers in Fedem: Axial- and joint springs and dampers. An axial spring or damper applies relative forces between two triads along the direction between the triads. Joint springs and dampers are integrated in the joint, and act on the joint triads along one of the joints unconstrained DOFs (see [Section 5.6.1, "Joint variables"](#)).

Both types of spring and dampers have the same options, and they will be described below. The joint springs and dampers are accessed through the Property Editor panel of the joint they are used in (see [Section 5.6.2, "Joint properties"](#)) while the axial springs and dampers are separate items with 3D symbols and their own Property Editor panels.

### 5.9.1 Spring properties

The spring properties (shown below) consist of the following options:



#### Stress free length/angle control

This group of options concerns the calculation of spring deflection. The deflection is defined as positive when it is increasing the spring length.

- ① *Length/Angle in model* – The current distance (for translation) or angle (for rotation) measured in the model as you have made it. For joint springs, this is the measured value of the joint DOF that the spring acts on. For axial springs, it is the distance between the two triads.
- ② *Initial stress free length/angle, Initial deflection* - These radio buttons and fields work together allowing you to introduce pre-stress in the spring, by setting an initial stress free length/angle different from the *Length/Angle in model*. You can chose to enter this property either as an absolute value, using the *Initial stress free length/angle* option, or relative to the *Length/angle in model* by selecting the *Initial deflection* option.



**CAUTION:** If you introduce a spring pre-stress in this manner, it will be accounted for in the very first iteration of the dynamics simulation and thus lead to a dynamic shock effect. However, when initial **Static equilibrium analysis** is switched on, the pre-stress force is taken as a pure static load and the transient shock should be avoided.



**NOTE:** The initial deflection is positive when it increases the spring length/angle.

- ③ Stress free length/angle change – You can select a function to change the stress free length/angle of the spring during the simulation (see [Section 5.11, "Functions"](#)). The value of the function will be used as an addition to the initial stress free length/angle defined above.



**TIP:** You can introduce motion into your system by using this option to change the length of a very stiff spring. However, an alternative and probably better way (for joint variables) is to use the Prescribed constraint type (see "[Joint variable properties](#)" in [Section 5.6.2](#)). In that case the stiff spring is avoided and the DOF is eliminated as an unknown from the system of equations. In most cases, this yields a more stable solution.

5

### Spring properties

This group of options controls how the deflection is evaluated to produce the spring force or torque. The spring can be either a linear spring with a constant stiffness, or a non-linear spring with a non-linear relationship between the deflection and the force/torque or the stiffness.

The spring force/torque is reckoned to be positive when it is working in the opposite direction of the increasing spring length/angle.

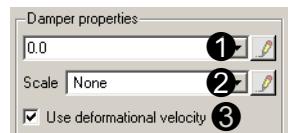
- ④ In this field you can enter a constant spring stiffness or select a defined spring characteristic from the pull-down menu.
- ⑤ Scale – The spring force or torque can be scaled by a function. This can for instance be used to switch the spring on and off during the simulation. When no function is selected the scale is set to 1.0.

### 5.9.2 Damper properties

This group of options controls the evaluation of a damper's force or torque from its velocity.

Both linear and non-linear dampers are allowed.

A linear damper uses a constant damping coefficient. A non-linear one uses a function to control how the damper force or coefficient depends on its velocity.



- ① In this field a constant damping coefficient can be entered, or you can select a damper characteristic from the pull-down menu.
- ② **Scale** – The damper force or torque can be scaled by a function. This can for instance be used to switch the damper on and off during the simulation. When no function is selected the scale is set to 1.0.



**NOTE:** In Fedem version 2.5m3 or lower, non-linear dampers were modeled using a function to change the damper coefficient. When opening such models in version 3.0 or higher, those dampers are converted by setting the damper functions as scale functions, and the coefficient to 1.0.

- ③ **Use deformational velocity** – This option is available only if the damper is acting together with a spring with a forced change in its stress free length. The option enables the usage of the deformational velocity of the connected spring, when evaluating the damper. The deformational velocity is the spring velocity without the velocity component coming from a forced change in the stress free length.



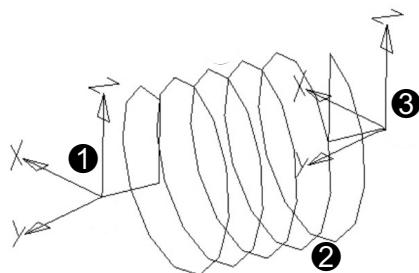
**NOTE:** The Use deformational velocity toggle is not visible for axial dampers unless there is a parallel axial spring connected to the same triads.

### 5.9.3 Axial spring symbol



The symbol for an axial spring is displayed in the *Modeler* view as shown to the right.

- ① First triad
- ② Axial spring
- ③ Second triad

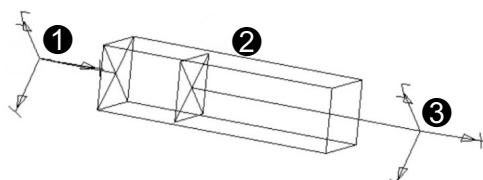


### 5.9.4 Axial damper symbol



The symbol for an axial damper is displayed in the *Modeler* view as shown to the right.

- ① First triad
- ② Axial damper
- ③ Second triad



## 5.9.5 Spring and damper characteristics

Non-linear springs and dampers are defined by creating a spring or damper characteristics objects. There are four basic types of characteristics available both for springs and dampers. The differences between them are whether they define a rotational or translational behavior, and whether they define a stiffness/damping coefficient curve or a force/torque curve. In addition, there are two types of advanced spring characteristics available (for translational and rotational springs, respectively), see [Section 5.9.6, "Advanced spring characteristics"](#).

### Spring characteristics

The four basic spring characteristics types are:

- Force - Translation
- Torque - Rotation
- Stiffness - Translation
- Stiffness - Rotation

*Force-Translation/Torque-Translation* - These characteristics describe the relationship between displacement and spring force/torque directly. The spring stiffness is then computed as the derivative of the provided curve.

$$k(\Delta) = \frac{dF(\Delta)}{d\Delta}$$

*Stiffness-Translation/Stiffness-Rotation* - These characteristics describe the relationship between the displacement and the spring stiffness directly. The spring force /torque is then computed as the integral of the provided curve from 0 to the current deflection:

$$F = \int_0^{\Delta} k(x) dx$$

The definition of the curves used can be done using one of the following function shape types (see [Section 5.11.5, "Function Types"](#)):

- Polyline and Polyline from file
- Constant
- Linear
- Ramp and Limited Ramp

A more detailed description of the spring characteristics can be found in the Fedem 7.6 Theory Guide, *Section 5.1, "Spring Elements."*

## Damper characteristics

The four types of damper characteristics are:

- Force - Velocity
- Torque - Angular velocity
- Coefficient - Velocity
- Coefficient - Angular velocity

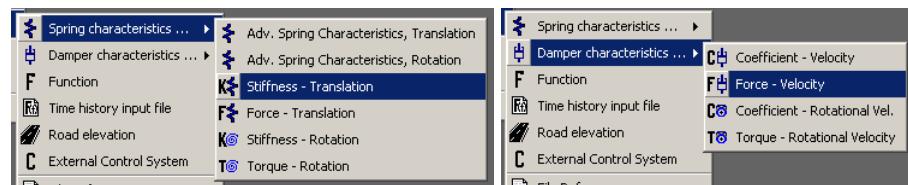
*Force-Velocity/Torque Angular velocity* - These characteristics describe the relationship between the damper velocity and damper force/torque directly. If a function  $g(v)$  is used, the damper force is  $F(v) = g(v)$  for all  $v$ . The damping coefficient is computed as the derivative  $g'(v)$ . A regular damper will have a  $g(v)$  that is positive for positive  $v$ , and vice versa.

*Coefficient-velocity/Coefficient-Angular velocity* - These characteristics are interpreted as the derivative of the force/torque-velocity function with respect to  $v$ . The damper force at a specific  $v$  is thus  $F(v) = \int_0^v g(w)dw$  for a given function  $g(v)$ , and the damping coefficient is the function value directly. A regular damper will have a coefficient-velocity function with positive values only.

The damper characteristics can be defined using the same function types as for the spring characteristics (see "[Spring characteristics](#)" above). A more detailed description of the damper characteristics can be found in the Fedem 7.6 Theory Guide, Section 5.2, "Damping Elements."

## Creating spring and damper characteristics

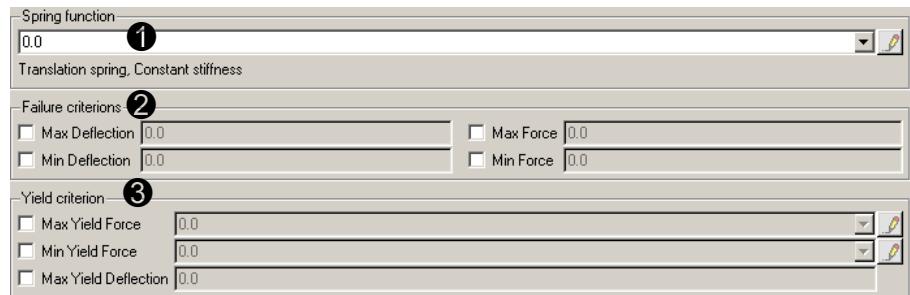
To create a spring or damper characteristic, right click in the Model Manager Objects list, and select **Create -> Spring/Damper characteristic** and then the type you want from the menus shown below.:



The characteristics you have created, will be displayed in the proper pull-down menus in the spring and damper Property Editor panels. Only those of correct type will be listed, to avoid using characteristics defined for rotation in translational DOFs and vice versa.

## 5.9.6 Advanced spring characteristics

In addition to the basic spring characteristics types described above, there are also some more advanced characteristics available with further options for defining the non-linear behavior of a spring. The Property Editor panel for the advanced spring characteristics is displayed below.



- ① Spring function** - In this field you may either enter a constant spring stiffness, or select an existing basic spring characteristics function from the pull-down menu.
- ② Failure criterions** - Failure of the spring can be defined through max and min. forces, and deflections. You can enable all four criteria, and whichever failure criterion is satisfied first will switch the spring (permanently) off (i.e., both spring force and stiffness vanishes).
  - *Max Deflection*: Spring is active until its deflection becomes greater than this value.
  - *Min Deflection*: Spring is active until its deflection becomes less than this value.
  - *Max Force*: Spring is active until force becomes greater than this value.
  - *Min Force*: Spring is active until force becomes less than this value.
- ③ Yield criterion** - Hysteresis behavior and/or permanent deflection after unloading can be introduced in springs by this options. The yield criterion will limit the force of the spring to the specified max and min. forces. When the spring force reaches any of these limits, the spring stiffness vanishes and any further deflection of the spring is defined as the *yield deflection*.
  - *Max Yield Force*: Spring force is always less than this value if this option is enabled.
  - *Min Yield Force*: Spring force is always greater than this value if this option is enabled.

- **Max Yield Deflection:** If the yield deflection exceeds this value (either on tension or compression) the spring is switched permanently off.



**Note:** The Max Yield Force and Min Yield Force can also be defined through functions giving the spring variable yield limits. This can be used to model "clutch-like" behaviors in a spring coupling, where you can smoothly (or abruptly) engage/disengage the motion coupling.

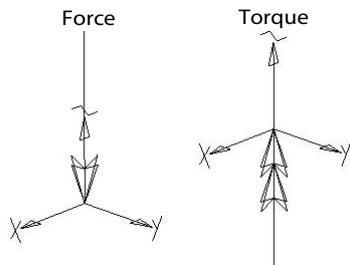
To define a non-linear spring with sudden failure and/or yield limits, you first have to define the non-linear elastic force-deflection (or stiffness-deflection) curve through the *Spring characteristics* menu (see "[Creating spring and damper characteristics](#)" above). Then you create an *Advanced Spring Characteristic* via the same menu, select the newly created Spring Characteristic in the *Spring function* pull-down menu, and then add the failure/yield criteria. The advanced spring characteristic is then available for selection in the *Spring properties* field of the Spring objects.

## 5.10 Loads



Two types of loads can be applied to triads or parts: forces and torques. Both types are applied as point-force vectors on FE nodes. These loads can be used to introduce motion into your mechanism. During simulation, the magnitude of forces and torques can be constant or controlled by functions (see [Section 5.11, "Functions"](#)). When creating them, it is possible to add a load directly to an existing triad or to an FE node.

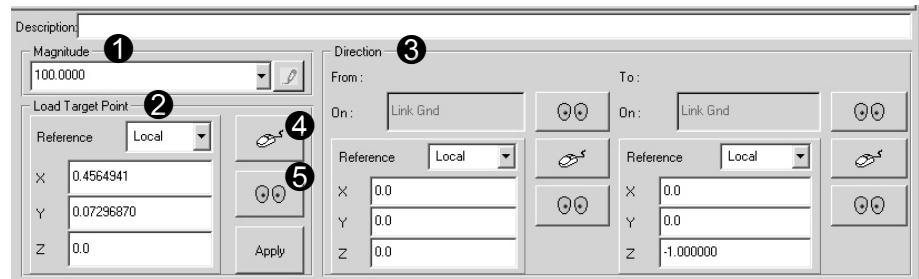
### 5.10.1 Load symbols



The symbols for forces and torques are displayed in the *Modeler* view as shown to the left.

## 5.10.2 Load properties

The magnitude and direction of a force/torque vector can be edited in the Property Editor panel (shown below). Select the force or torque to show its properties.



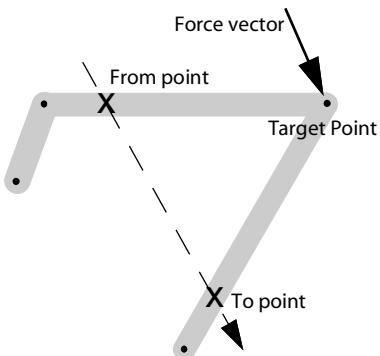
- ➊ *Magnitude* – to change the load magnitude, enter a constant value or select one from the list of functions in your model (see [Section 5.11, "Functions"](#)).
- ➋ *Load Target Point* – the point is given in either global or local coordinates. You can also select a new target point using the **Mouse** button (see [Section 5.10.3, "Target point"](#) below).
- ➌ *Direction* – The *From* and *To* options allow you to edit the orientation of the load vector (see [Section 5.10.4, "Direction"](#) below).
- ➍ You can select a new point for the *Load Target Point*, or *From* or *To Directions* using the **Mouse** buttons (then select a new point in the *Modeler* view).
- ➎ You can click and hold down any of the **View** buttons in the Property Editor panel to highlight the corresponding point in the *Modeler* view.

## 5.10.3 Target point

To specify a new target point, click the **Mouse** button and use the cursor to select a part in the *Modeler* view. If the target point does not coincide with an FE node, the target point will snap to the closest node. Press **Done** to confirm the selection. A triad is created at that position.

### 5.10.4 Direction

The direction of the input load vector can be specified by two points moving together with the selected parts, or by two fixed points given in global coordinates. The direction is given by the vector pointing from the *From point* to the *To point* (shown at right).



## 5.11 Functions

### F

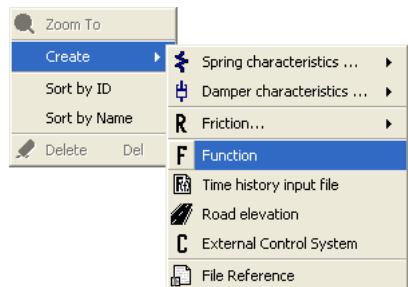
Functions can be used to control the magnitude of loads, the length of springs, prescribed motion in joints, etc. The function defines one input variable (except for functions of type *Math Expression*, which may have up to four input variables) and a function shape that is used to transform the input value(s) into the output value of the function. The output will thus change during the simulation depending on the variations in its input value(s).

The input value can be a system variable measured by a sensor, the output of a control system, the output of a different function, or simply the simulation time. The function shape can be defined in several different ways, and uses a common way of defining function shapes across different objects needing to do so.

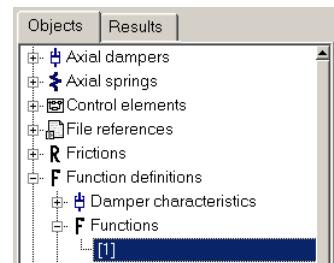
Road elevations, Control inputs, Control outputs, Spring and Damper characteristics are all examples of objects using a similar way of defining function-like relationships. The description found here is thus valid for several other objects as well.

### 5.11.1 Creating a function

You create a function by right-clicking an empty space in the Model Manager *Objects* list, selecting **Create** and then **Function**. You can also access the command from the *Mechanism* menu in the main window.



The new Function is automatically selected, and its properties are shown in the Property Editor panel. It will also be added to the list of Functions maintained in the Model Manager *Objects* list (shown at right).



5

### 5.11.2 Function properties

When a function is selected in the *Objects* list, its properties are displayed in the Property Editor panel (shown below) which is divided in three parts. The left part contains fields for defining the function type and argument and the middle part contains a list of parameters associated with the chosen function type. The right part contains two tabs; one for displaying an image explaining the function definition and another with options for previewing the function in a graph.



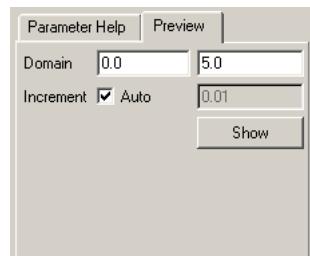
- ① **Function Type** - You can change the function type by selecting the new type from the pull-down menu. The parameters and the help image shown in the panel are updated to reflect the new type.
- ② **Argument** - You can select any of the objects in your model that already is used as an argument by this or another function; namely

those having a sensor attached, from the pull-down menu. When an object has been selected, specify which quantity you want to access on that object by selecting from the *DOF* and *Var* pull-down menus.

- ③ **Selection button** - By pressing this button you can select any object in your model to use as argument. When the button is pressed, the Guide panel will prompt you to select an object. Do so and press **Done** to accept the selection. A sensor will then be created on the selected object which will appear in the *Argument* pull-down menu. In addition to physical objects like Triads, Joints, etc., you may also select Control output elements and other Functions as arguments.
- ④ **Parameters** - This frame contains the user-defined parameters of the selected function type.
- ⑤ **Parameter Help** - This tab displays a reference picture to easier remember the meaning of the different parameters.
- ⑥ **Preview** - This tab has options to control preview of the function shape (see [Section 5.11.3, "Preview"](#)).

### 5.11.3 Preview

To get an impression of the function shape you may preview it as a curve in a graph. Specify the argument *Domain* and *Increment* in the *Preview* tab of the Property Editor panel, then push the **Show** button to plot. A preview graph containing the preview curve is then created and displayed. The displayed curve is updated automatically when changing any of the function properties.



Most functions have the option to set the preview increment automatically. This is enabled by default, but can be disabled by toggling the *Auto* toggle.

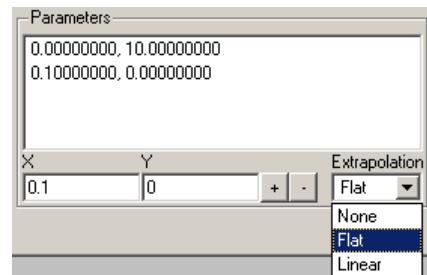
The created graph and curve is automatically added to the Model Manager Results list and may be handled like any regular graph and curve. (Refer to [Section 9.1, "Postprocessing environment"](#) and [Section 9.2, "Graphs"](#).)



**TIP:** The actual values of a function can also be plotted directly in a Graph when you have run a simulation or during a simulation. To enable such plotting, you must first specify the Additional Solver Option `-allEngineVars` for the Dynamics Solver (see [Section 8.2, "Additional solver options"](#)) before starting the simulation, such that the function values are saved to the results database files for the computed time steps.

## 5.11.4 Extrapolation

For functions defined on a user-specified finite domain, the option to extrapolate the function outside this domain exists (i.e., functions of type Polyline, Linear derivative or Spline). The default is no extrapolation (None).



If the *Extrapolation* option is set to Flat the function retains the end point values when outside its domain. That is:

- For all  $v < x_1$  the function evaluates to  $f(x_1)$ .
- For all  $v > x_n$  the function evaluates to  $f(x_n)$ .

If the *Extrapolation* option is set to Linear the function is continued along the tangent line of the nearest end point, that is:

- For all  $v < x_1$  the function evaluates to  $f(x_1) + f'(x_1) \cdot (v - x_1)$ .
- For all  $v > x_n$  the function evaluates to  $f(x_n) + f'(x_n) \cdot (v - x_n)$ .

5

## 5.11.5 Function Types

All function types available in Fedem are presented in the following.

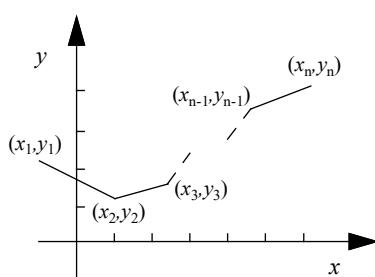
### 1:1

This represents the identity function. Its value equals the argument value.

### Polyline

This is linear interpolation between user-specified points  $(x_i, y_i)$ .

$$f(v) = y_i + \frac{v - x_i}{x_{i+1} - x_i} (y_{i+1} - y_i) , x_i < v \leq x_{i+1}, x_i, y_i, i \in [1, n]$$

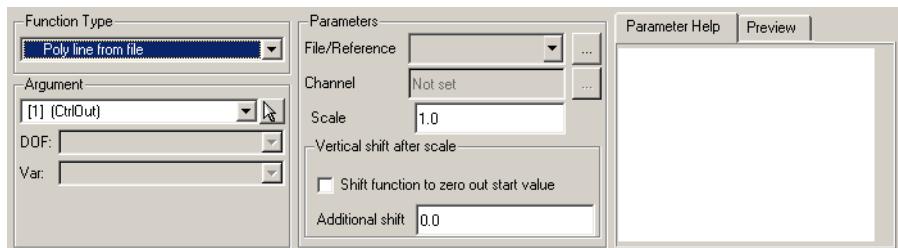


To add many numbers to a polyline function, copy and paste is feasible. Copy the numbers from the application where they are present (e.g. a spreadsheet or a text editor). On win32 based systems they have to be copied to the clip board by pressing ***Ctrl+C***. The numbers are then inserted into the polyline function by clicking the list and pressing ***Ctrl+V***.

Refer to [Section 5.11.4, "Extrapolation"](#) to learn about the extrapolation of a polyline function.

### Polyline from file

This function is similar to the polyline function except that the points now are read from a file. The file format can either be single- or multi-channel ASCII (***.asc***, ***.txt***), DAC (***.dac***) or RPCIII time history (***.rsp***, ***.drv***, ***.tim***). For multi-channel ASCII and RPCIII files, the channel to read is chosen by pressing the select button [...] . As shown below, additional parameters for scaling and vertical shift of the function may also be specified in the Property Editor panel of this function type.



For abscissa value  $v$  the returned ordinate  $f(v)$  of the polyline from file function is

$$f(v) = \left( y_i + \frac{v - x_i}{x_{i+1} - x_i} (y_{i+1} - y_i) \right) s + k \quad , x_i < v \leq x_{i+1} , x_i, y_i, i \in [1, n]$$

$s$  - scale

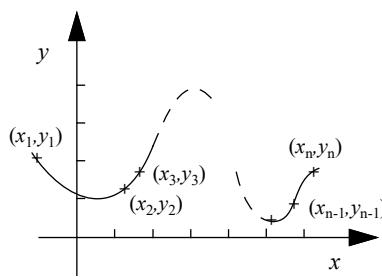
$k$  - vertical shift

This function interpolates linearly between the user-specified points  $(x_i, y_i)$ . The interpolated ordinate value is then scaled by a factor  $s$ , and shifted by  $k$ . The vertical shift value  $k$  is set to the scalar  $\kappa$  entered in the *Additional shift* field. If *Shift function to zero out start value* is checked as well, the start value  $x_1$  is also subtracted, so that  $k = \kappa - x_1$ .

For abscissa values outside the function domain an extrapolated ordinate is assigned, such that

$$v < x_1 \Rightarrow f(v) = f(x_1) \text{ and } v > x_n \Rightarrow f(v) = f(x_n)$$

### Spline



$$x_i, y_i, i \in [1, n]$$

A third order spline approximation is calculated from a set of user-specified points  $(x_i, y_i)$ , which may be entered in the same way as for Polyline functions. At least 4 points are required.

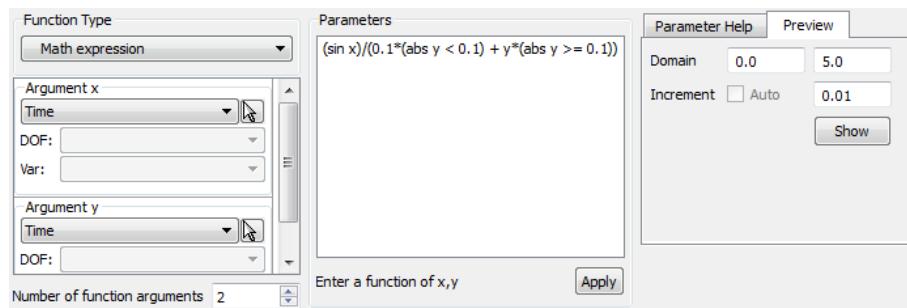
Refer to [Section 5.11.4, "Extrapolation"](#) to learn about the extrapolation of a Spline function.



**TIP:** If you have a [Polyline](#) function consisting of at least four points, you can change its type into Spline without loosing the entered curve point data. This is useful just to see how the same set of points appear when they are interpolated with a cubic spline basis instead of the piece-wise linear interpolation. You can then change back to Polyline again, if you want to retain the linear interpolation in the simulation.

### Math Expression

The math expression function type allows a closed form function expression to be entered as free text. This is the only function type that may have more than one input variable (function arguments). The function expression is entered in the Parameters section of the Property Editor panel as shown below. The **Apply** button must be pushed to check and register the expression, whenever it is updated. The number of input variables are also set in the Property Editor panel (the default is one input variable).



Rules for Fedem math expressions:

- The expression must be functions of either one, two, three or four variables, and the independent variable must be named  $x, y, z$  and  $t$ .
- The expression may consist of the intrinsic functions and operators listed below, along with the independent variables, { $x, y, z, t$ }.
- Function expressions may be nested, e.g:
  - $\sin(\cos x)$  - is a valid expression.
- Precedence is set by parentheses in the usual manner. E.g:
  - $\sin(x^2)$  -  $x$  is squared before being input to the sine function.
  - $(\sin x)^2$  - the value of sine of  $x$  is squared.
- The logical operators have a return value of 0 if *FALSE* and 1 if *TRUE*, and are used by multiplying them with the function expressions. E.g:
  - $\sin x + (x>2)*x$  equals  $(\sin x)$  if  $(x \leq 2)$ , and  $(\sin x + x)$  if  $(x>2)$ .

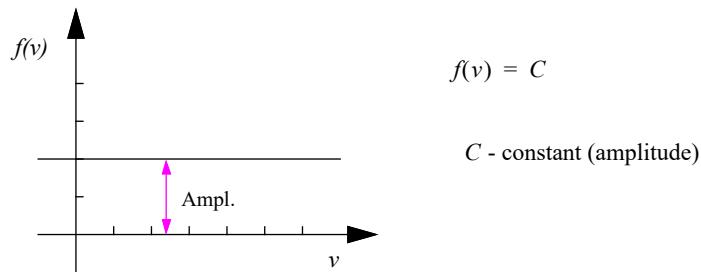
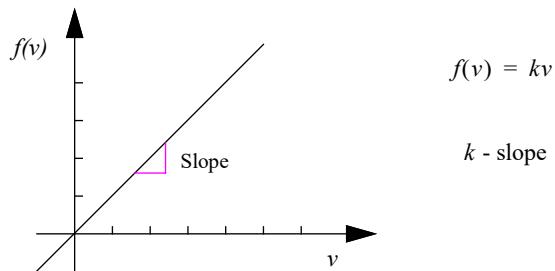
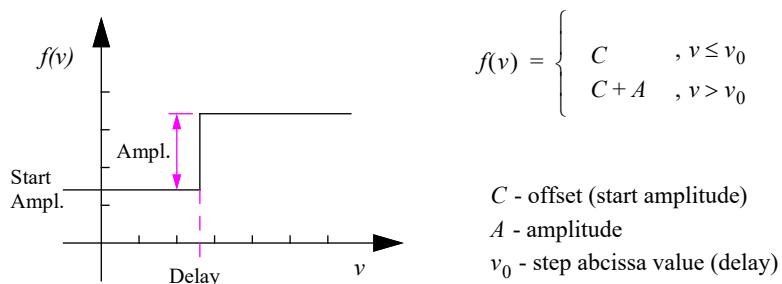
Below are the available intrinsic functions and operators. The symbols  $a$  and  $b$  may both be numbers, functions or the independent variable  $x$ .

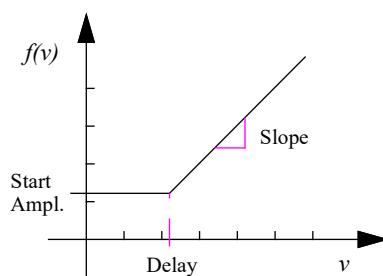
Function	Description
$abs\ a$	absolute value
$a+b$	addition
$a-b$	subtraction
$a*b$	multiplication
$a/b$	division
$a \% b$	modulus
$a ^ b$	power

Function	Description
$\max(a, b)$	Maximum of $a$ and $b$
$\min(a, b)$	Minimum of $a$ and $b$
$\sqrt{a}$	square root
$n\#a$	$n^{\text{th}}$ root
$\sin a$	sine
$\cos a$	cosine
$\tan a$	tangent
$\arcsin a$	arcsin
$\arccos a$	arccos
$\arctan a$	arctan
$\ln a$	logarithm, base e
$\log a$	logarithm, base 10
$\exp a$	exponential, $e^a$
$aEb$	$a \cdot 10^b$
$\pi$	constant value of $\pi$

Available logical operators:

Function	Description
$!a$	not
$a < b$	less than
$a > b$	greater than
$a \leq b$	less than or equal to
$a \geq b$	greater than or equal to
$a == b$	equal to
$a != b$	not equal to
$a    b$	or
$a \&& b$	and

**Constant****Linear function****Step**

**Ramp**

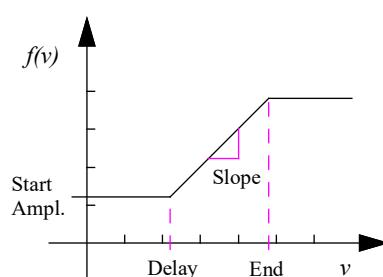
$$f(v) = \begin{cases} C & , v \leq v_0 \\ C + k(v - v_0) & , v > v_0 \end{cases}$$

$C$  - offset (start amplitude)

$k$  - slope

$v_0$  - ramp start abscissa value(delay)

**5**

**Limited ramp**

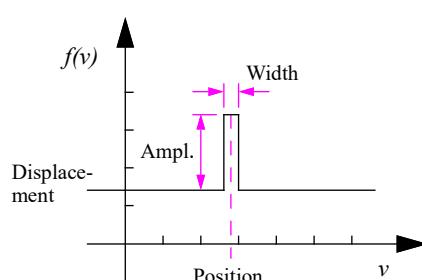
$$f(v) = \begin{cases} C & , v \leq v_0 \\ C + k(v - v_0) & , v_0 < v < v_e \\ C + k(v_e - v) & , v \geq v_e \end{cases}$$

$C$  - offset (start amplitude)

$k$  - slope

$v_0$  - ramp start abscissa value(delay)

$v_e$  - ramp end abscissa value(end)

**Pulse**

$$f(v) = \begin{cases} C & , v \leq v_0 - \frac{\delta}{2} \\ C + A & , v_0 - \frac{\delta}{2} < v < v_0 + \frac{\delta}{2} \\ C & , v \geq v_0 + \frac{\delta}{2} \end{cases}$$

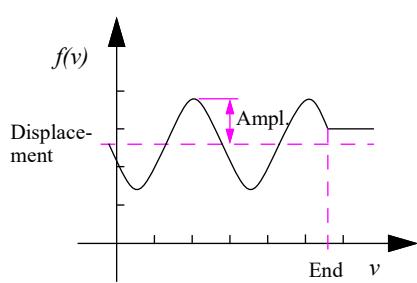
$C$  - offset (displacement)

$A$  - amplitude

$\delta$  - width

$v_0$  - pulse center abscissa value (position)

### Sine



$$f(v) = \begin{cases} C + A \sin(2\pi(f_0v - \theta)) & , v \leq v_e \\ C + A \sin(2\pi(f_0v_e - \theta)) & , v > v_e \end{cases}$$

$C$  - offset (displacement)

$A$  - amplitude

$f_0$  - frequency (Hz)

$\theta$  - phase shift (fraction of period)

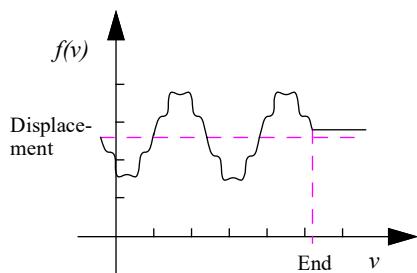
$v_e$  - end of sinusoidal



**NOTE:** If the End value in a Sine function is specified less than or equal to zero, that is interpreted as infinity.

### Combined sine

$$f(v) = \begin{cases} C + A_1 \sin(2\pi(f_1v - \theta_1)) + A_2 \sin(2\pi(f_2v - \theta_2)) & , v \leq v_e \\ C + A_1 \sin(2\pi(f_1v_e - \theta_1)) + A_2 \sin(2\pi(f_2v_e - \theta_2)) & , v > v_e \end{cases}$$



$C$  - offset (displacement)

$A_1, A_2$  - amplitude

$f_1, f_2$  - frequency (Hz)

$\theta_1, \theta_2$  - phase shift (fraction of period)

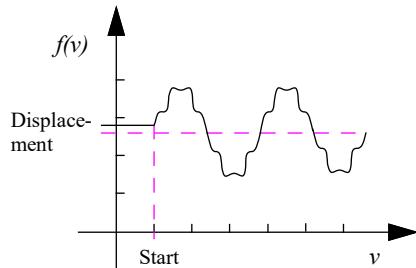
$v_e$  - end of sinusoidal



**NOTE:** If the End value in a Combined sine function is specified less than or equal to zero, that is interpreted as infinity.

### Delayed combined sine

$$f(v) = \begin{cases} C + A_1 \sin(2\pi(f_1 v_0 - \theta_1)) + A_2 \sin(2\pi(f_2 v_0 - \theta_2)), & v \leq v_0 \\ C + A_1 \sin(2\pi(f_1 v - \theta_1)) + A_2 \sin(2\pi(f_2 v - \theta_2)), & v > v_0 \end{cases}$$



$C$  - offset (mean value)

$A_1, A_2$  - amplitude

$f_1, f_2$  - frequency (Hz)

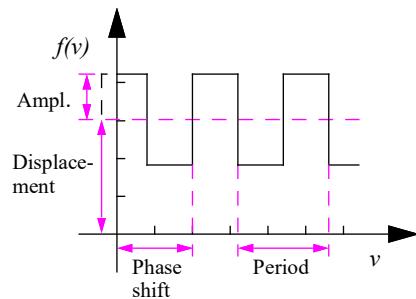
$\theta_1, \theta_2$  - phase shift (fraction of period)

$v_0$  - delay

5

### Periodic square pulse

$$f(v) = \begin{cases} C + A, & \frac{n-1}{f_0} < v - \theta \leq \frac{2n-1}{2f_0} \\ C - A, & \frac{2n-1}{2f_0} < v - \theta \leq \frac{n}{f_0} \end{cases} \quad n \in 0, 1, 2, \dots$$



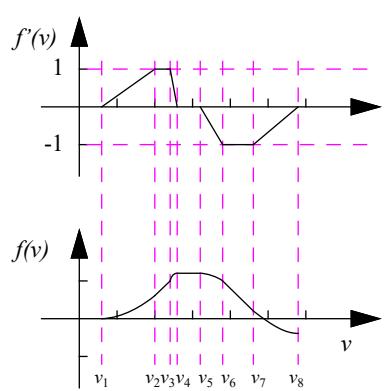
$C$  - offset (displacement)

$A$  - amplitude

$f_0$  - frequency(inHz when  $v$  is time)

$\theta$  - phase shift

### Linear derivative function



$$f'(v) = \begin{cases} 0 & , 0 \leq v < v_1 \\ \frac{v - v_1}{v_2 - v_1} & , v_1 \leq v < v_2 \\ 1 & , v_2 \leq v < v_3 \\ \frac{v_4 - v}{v_4 - v_3} & , v_3 \leq v < v_4 \\ 0 & , v_4 \leq v < v_5 \\ \frac{v_5 - v}{v_6 - v_5} & , v_5 \leq v < v_6 \\ -1 & , v_6 \leq v < v_7 \\ \frac{v - v_8}{v_8 - v_7} & , v_7 \leq v < v_8 \\ 0 & , v_8 \leq v < v_9 \\ \frac{v - v_9}{v_{10} - v_9} & , v_9 \leq v < v_{10} \\ \vdots \end{cases}$$

$$f(v) = \int_0^v f'(\tau) d\tau$$

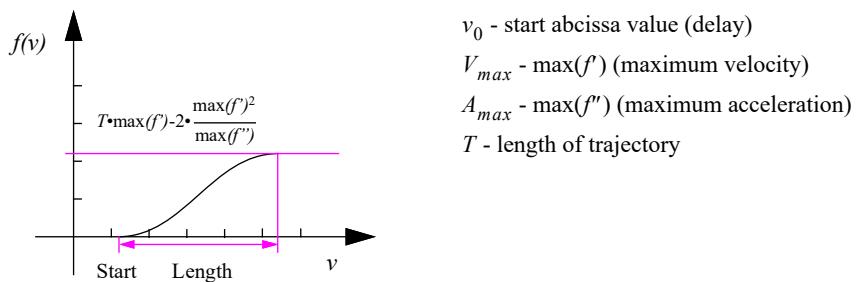
To add many numbers to a linear derivative function, copy and paste is feasible. Copy the numbers from the application where they are present. (e.g. a spreadsheet or a text editor). On win32 based systems they have to be copied to the clip board by pressing *Ctrl+C*. The numbers are then inserted into the polyline function by clicking the list and pressing *Ctrl+V*.

Refer to [Section 5.11.4, "Extrapolation"](#) to learn about the extrapolation of a Linear derivative function.

### Smooth trajectory

$$f(v) = \begin{cases} 0 & , t \leq 0 \\ \frac{A_{max}}{4} \left( t^2 - \frac{1}{2\omega^2} (1 - \cos(2\omega t)) \right) & , 0 < t \leq \frac{\pi}{\omega} \\ V_{max} \left( t - \frac{\pi}{2\omega} \right) & , \frac{\pi}{\omega} < t \leq T - \frac{\pi}{\omega} \\ V_{max} \left( T - \frac{\pi}{\omega} \right) - \frac{A_{max}}{4} \left( (T-t)^2 - \frac{1}{2\omega^2} (1 - \cos(2\omega(T-t))) \right) & , T - \frac{\pi}{\omega} < t \leq T \\ V_{max} \left( T - \frac{\pi}{\omega} \right) & , t > T \end{cases}$$

where  $t \equiv v - v_0$  and  $(\omega \equiv \frac{\pi A_{max}}{2 V_{max}})$



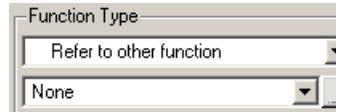
### External function

This function type have no argument and no properties, except for a *Channel* index which is incremented for each instance of this function type. External functions are used for passing external input to a running dynamics simulation managed by scripting, e.g., by using the *fedempy* python interface.

Alternatively, the function values can be read from a specified file, see [Section 4.9.5, "External function values from file".](#)

### Refer to other function

This function type reuses a previously defined function shape within this function. When selected, a drop-down menu (shown at right)



appears from which you can select the function you want to reuse the function shape definition from. The definitions become linked, and the changes you make to the referenced function will also be reflected in the referring function.

### User-defined functions

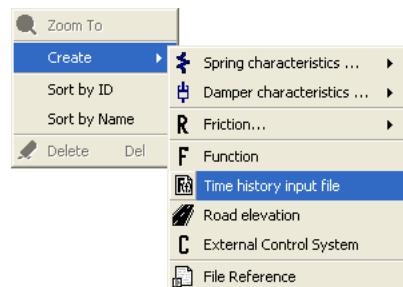
You may also implement your own set of multi-argument user-defined function types, using the plug-in capability of Fedem. See [Section 2.11, "Customizing Fedem using plug-ins"](#) for information on doing that.

## 5.11.6 Time history input files

A special type of function definitions are the *Time history input file* object. This object behaves essentially as a function of time, but is optimized to be used for input of time history data from an external files.

### Creating

To create a *Time history input file* object, right click in the Model Manager *Objects* list, select **Create** and **Time history input file**. A new object will then be created and its properties are displayed in the Property Editor panel.



### Properties

The behavior and options for this object is the same as for the [Polyline from file](#) function type, except that it is always a function of time. See [Section 5.11.5, "Function Types"](#).

## 5.12 Sensors

*Sensors* are used to measure movement and other variables associated with mechanism elements during the dynamics simulation. They work mostly as Tags to show what objects are being measured. Objects that have a sensor attached will appear in the *Argument* drop-down menu in *Functions* and *Control Inputs*. They are created automatically when selecting an *Argument* by using the selection button in the *Function* and *Control Input* Property Editor panels. See [Section 5.11.2, "Function properties"](#) and [Section 7.2, "Input and output"](#). They can also be created manually using the **Simple Sensor** and **Relative Sensor** commands.

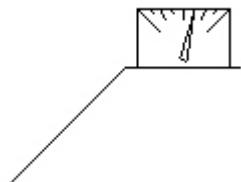
Among the measured quantities are triad positions and joint variables, the associated velocities and accelerations, triad forces, spring/damper forces, lengths and deflections/velocities, etc. Triad rotations can also be measured in terms of Euler-Z-Y-X angles in the global coordinate system, or relative to another Triad with relative sensors. The data obtained from a sensor can be processed by a function and used in the model or a control system (see [Section 5.11, "Functions"](#) and [Chapter 7, "Control System Modeling"](#)).

There are two types of sensors, simple sensors and relative sensors:

### 5.12.1 Simple sensors



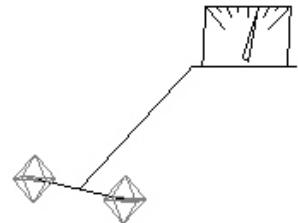
*Simple sensors* are used to tag a single mechanism element, such as a joint or triad, in order to extract measurements from it. When an object is tagged with a sensor, it will appear in the *Argument* drop-down menu in Functions and the *Input* menu in Control Inputs. In addition a 3D symbol will be created to show that this particular object is being measured. The symbol for a simple sensor is displayed in the *Modeler* view as shown to the right.



### 5.12.2 Relative sensors

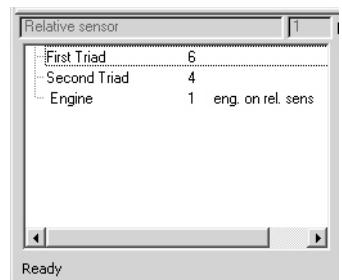


*Relative sensors* are used to tag two triads in order to extract relative measurements from them. The sensor will be displayed in the *Argument* drop-down menu in Functions and the *Input* menu in Control Inputs. The symbol for a relative sensor is displayed in the *Modeler* view as shown to the right.



### 5.12.3 Managing sensors

Sensors are managed in the Model Manager *Object* list. If you have created sensors, you can expand the *Sensors* group to view the list of sensors in your model. When a sensor is selected in the *Objects* lists, it is highlighted in the *Modeler* view and its description is displayed in the Property Editor panel. The ID and Topology panel (shown at right) shows the triad(s) to which the sensor is attached and lists the Functions or Control input elements using it.



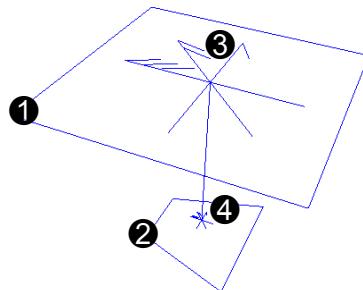
## 5.13 Strain rosettes



Strain rosettes are used to recover stresses and strains on a particular spot on your mechanism. The output is similar to the output from real strain gages in addition to standard strain and stress quantities like Von Mises, principal stresses/strains, and angle of max/min. principals. They can be placed on any of the FE models in the mechanism both before and after solving the dynamics. The strain rosette recovery is done by the strain rosette analysis (see [Section 8.8, "Strain rosette analysis"](#)), which recovers the strain rosettes data for one FE model at a time. See also the Fedem 7.6 Theory Guide, [Section 9.5 "Virtual strain gauges"](#) for the theoretical basis of strain rosettes.

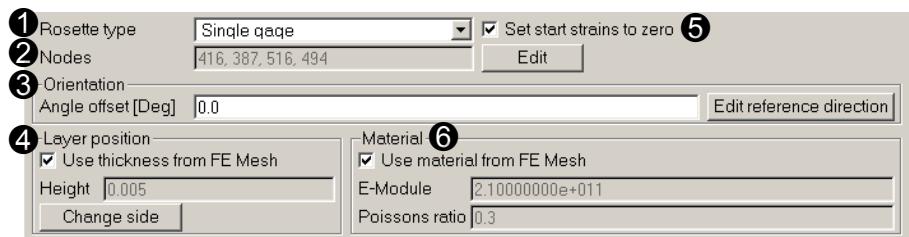
Strain rosettes are defined by selecting 3 or 4 FE nodes and the direction of the first leg of the rosette. The nodes defines a strain element in which the strains and stresses is calculated at the centroid of the element based on the deformation of the nodes. Material properties and shell thickness are extracted from the underlying FE mesh, but can be overridden manually.

As the picture to the right indicates, the strain rosette symbol consists of different parts, and visualizes several aspects of the virtual strain rosette.



- ① An enlarged symbol of the strain rosette to make it easier to find.
- ② Lines connecting the FE nodes used, showing a wireframe of the strain element.
- ③ Enlarged arrows showing the directions of the particular legs within the strain rosette. Leg 1, 2 and 3 is distinguished by the number of lines used to draw the head of the arrows.
- ④ A small symbol showing the exact position of the virtual strain rosette. The position shown also includes the shell thickness or layer position, and will thus often be above the strain element wireframe.

The different features of the strain rosettes can be accessed via the Property Editor panel, as shown below.



- 1** *Rosette type* – Menu to select between different rosette configurations. Single Gage, Double gage 90, Triple Gage 60 and Triple Gage 45.
- 2** *Nodes* – This field displays the FE node numbers used by the strain gage. The **Edit** button lets you select a different set of nodes.
- 3** *Orientation* – The orientation of the strain rosette is calculated from a reference direction and an angle offset. The reference direction can be selected by pressing the **Edit reference direction** button. You can select either an edge, or two points. The angle offset is entered in the field as degrees.
- 4** *Layer position* – This is a feature that applies to strain rosettes on shell elements. The strains and stresses will vary through the thickness of the shell elements, with the extreme values on the top and bottom. Normally, when *Use Thickness from FE Mesh* is toggled, the strains are calculated at either the top or bottom. The *Height* field will then display the position of the strain rosette above the mid-plane of the shell element, e.g. half the thickness. The **Change side** button can be used to switch the position of the strain rosette from one side to the other. To calculate the strains and stresses at another level through the thickness of the shell element, toggle off *Use Thickness from FE Mesh* and enter the new position above the element mid-plane.
- 5** *Set start strains to zero* – This toggle will zero out the strains at the first time step, eliminating strains due to gravitation or pre-stress effects when using the optional initial equilibrium iteration (see [Section 8.1.2, "Dynamics analysis"](#) and [Section 8.5.2, "Dynamics Solver \(Advanced Mode\)"](#)).
- 6** *Material* – The stress calculations will normally use the material data from the FE mesh. If those values are inadequate for some reason, they can be overridden by toggling off the *Use material from FE Mesh* toggle, and enter proper values for E-Module and Poisson's ratio.

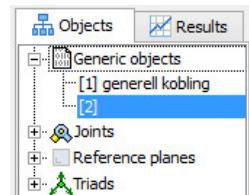
## 5.14 Generic database objects

A generic database object (not to be confused with *Generic Parts*) can be used represent simulation model objects that does not yet have their own object type in the Fedem graphical user-interface.



A generic database object is created by selecting **Generic DB Object** from the *Mechanism* menu.

They have no visualization in the 3D view, but a node under the *Generic objects* heading will appear in the *Objects* list of the Model Manager panel, as shown to the right. The Property Editor panel of an generic database object is shown below.



- ➊ **Type** – The string in this field is the keyword identifying the object type for the dynamics solver.
- ➋ **Definition** – In this field you enter the object definition itself, using the syntax of the solver input file (.fsi). If this field contains references to other objects in the model (as in the example above), the ID numbers used are the base ID of those objects. (You can find the base ID of an object using the Object Browser, see [Section 2.4.6, "Object Browser"](#).)



**CAUTION:** The contents of the *Definition* field is parsed directly by the dynamics solver. It is therefore imperative that the data specified here have the correct syntax and are otherwise valid, or else the dynamics solver may fail. Generic database objects should therefore only be used with guidance from Fedem support personnel.

## 5.15 Sub-assemblies

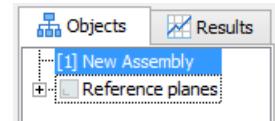
A Fedem model normally consists of a large number of objects of the different mechanism element types described in the previous section of this chapter, together with their properties and topological relations. However, as the model grows more and more complex, it is often convenient to group the various parts of the model into sub-models, or sub-assemblies. For this purpose, we have introduced the sub-assembly as a separate mechanism element in Fedem.

A sub-assembly is a meta-object in Fedem, and may consist of any number of the basis element types, described previously in this chapter. A sub-assembly may also contain other sub-assemblies as members, thereby creating a nested conceptual model.

### 5.15.1 Creating sub-assemblies

A new, empty sub-assembly can be created at any time during a modeling session, by selecting **Create -> Subassembly...** from the right-click menu in the *Objects* list of the Model Manager panel, or by selecting the **Subassembly...** item from the *Mechanism* menu.

A new node with description "New Assembly" representing the created sub-assembly then appears in the *Objects* list of the Model Manager panel, as shown to the right. This node is also highlighted indicating it is automatically selected.

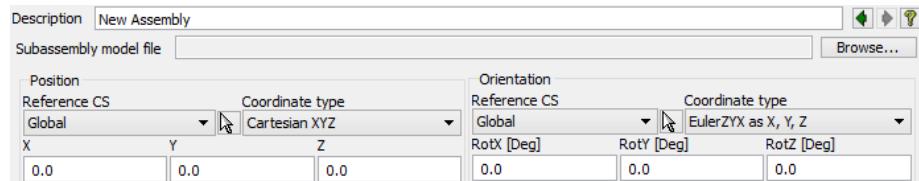


To populate the new sub-assembly with elements (Triads, Parts, Beams, etc.) you only need to keep the sub-assembly highlighted (selected) while performing the commands creating the other objects.

You can also create a sub-assembly from a set of existing mechanism objects. Simply multi-select the objects that you want to put in an assembly from the *Objects* list of the Model Manager panel, before using the **Create -> Subassembly...** command as described above. The selected objects will then be moved to the newly created sub-assembly.

### 5.15.2 Sub-assembly properties

In its general form, a sub-assembly object is only a grouping of other mechanism elements and has no other properties except for an associated name of an *Subassembly model file*, and fields for *Position* and *Orientation*, as shown in the Property Editor panel below.



A separate model file may be assigned to the selected sub-assembly by using the **Browse...** button. In that case, all the elements of that assembly are stored in the specified file when the model is saved. It is then possible to reuse this sub-assembly in another model by importing it into an

existing model (see [Section 5.15.3, "Importing sub-assemblies"](#)). If the *Subassembly model file* field is empty, the elements of the sub-assembly are stored in the main model file together with the rest of the model.

The *Position* and *Orientation* fields are used to display and edit the position and orientation of the whole sub-assembly. These fields work in the same way as those in the *Origin* tab of Triads, Parts and Joints (see [Section 4.5.4, "Origin property"](#) for details). The position and orientation of all mechanism elements within a sub-assembly is stored as the relative position with respect to the sub-assembly itself. Therefore, when the position and/or orientation of a sub-assembly is changed, all its mechanism elements automatically are moved along with the assembly.



**NOTE:** A sub-assembly can be moved only if none of its Triads or Joints (except for Cam Joints and Free Joints) are connected to elements in other sub-assemblies. Neither can it be moved if any of its Triads or Joints are attached to ground.

### Sub-assembly specializations

Various specializations exists of sub-assemblies, representing certain higher-level components in Wind Power and Marine modeling. These sub-assembly objects have different fields in their Property Editor panels depending on the actual assembly type, but they all include a *Description* and a *Mass* field. The latter displays the total weight of all the components of the assembly. Another common property is the *Visualize 3D* toggle. This turns on and off the 3D-visualization of all beams contained in the sub-assembly.

See the chapters on wind power and marine modeling for a closer look at the Property Editor panels of the sub-assemblies used in such structures.

## 5.15.3 Importing sub-assemblies

If a sub-assembly has been saved in a separate model file by assigning it a file name in the Property Editor panel (see [Section 5.15.2, "Sub-assembly properties"](#)), that sub-assembly may be imported into an existing model by selecting **Import Subassembly...** from the *File* menu. All the elements stored in that file are then copied into the current model, retaining their properties and inter-element connections. However, the original global position of the assembly is not retained. Instead the position/orientation of the assembly is reset to identity such that you have to reposition it manually after the import. The original user ID of the imported sub-assembly is retained, unless it conflicts with another sub-assembly having the same user ID. In that case it is automatically assigned a new user ID instead.



**WARNING!** It is not recommended to import sub-assemblies which also contains objects with references to other objects not part of the same assembly. In that case, these references will be broken such that the imported model will not be consistent and cannot be solved before the broken connections are fixed manually.

The **Import Subassembly...** command can be used on any other Fedem model file - not only files containing sub-assemblies. This way, any existing Fedem model may be imported as a part of a new model, enabling a hierarchical way of modeling complex mechanisms and reusing models that has been used before in a larger setting.



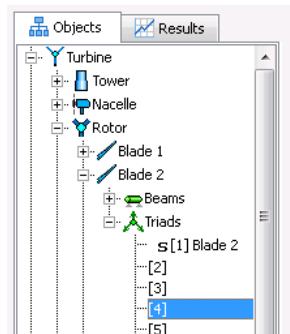
**NOTE:** When a normal model file is imported as a sub-assembly, only the structural objects in that file is imported. All result objects (Graphs, Curves and Animations) are ignored. Also simulation event definitions, if any, are ignored, as well as general visualization settings and model preferences.

#### 5.15.4 User ID convention in assemblies

The user ID numbers of a given object type is unique only within each sub-assembly. Thus, there may exist many, e. g., triads with user ID 1, etc., in a given model consisting of several sub-assemblies. To uniquely identify the different objects of such models, a composite value on the format [UID,AID1,...,AIDn] is used. The convention here is that each ID from left to right in the brackets brings you one step further up the hierarchy. So the first value is the user ID of the particular object, next is the user ID of the sub-assembly containing that object, then the user ID of the sub-assembly containing the first sub-assembly, and so on.

In the image to the right, the selected triad has user ID 4 in the "Blade 2" sub-assembly. This blade has user ID 2 in the "Rotor" sub-assembly, which again has user ID 3 in the "Turbine" sub-assembly, which has user ID 1. The total composite ID for this triad is then [4,2,3,1].

The composite user ID is used in the Model manager, *Topology* view and *Output List* view, and also in the direct output from the dynamics solver in the `fedem_solver.res` file.



**CAUTION:** The composite user ID is not used in pull-down menus when a reference to some existing mechanism element is to be selected. In those cases it can therefore be difficult to distinguish various objects of same type and with common user ID, unless they have been assigned a unique description.



# Chapter 6 Marine Modeling

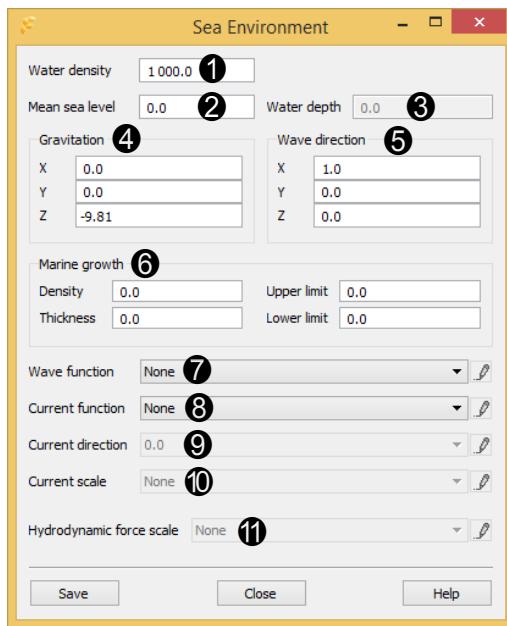
This chapter describes some additional features in Fedem, that enables the modeling of sea environment and load conditions due to the effect of the sea. These features make it possible to model floating mechanisms, and mechanisms that are partly or fully submerged in sea water.

Sections in this chapter address the following topics:

- Sea environment
- Wave and current functions
- Response amplitude operators
- Beam string structures
- Space frame structures
- Soil Pile structures
- Hydrodynamic load calculations

## 6.1 Sea environment

The Sea Environment dialog box (shown below) is accessed from the *Marine* menu in the Fedem main window. It is used to specify the global sea environment properties that are used in the calculation of Hydrodynamic loads on the Fedem mechanism.



- ① **Water density** - Specifies the mass density of the sea water.
- ② **Mean sea level** - Specifies the mean sea water level (MSL). That is, the distance from the global origin to the sea surface at calm sea, in the opposite direction of the gravitation vector.
- ③ **Water depth** - Specifies the water depth. That is, the distance from the calm sea surface to the sea bed, in the direction of the gravitation vector. This value affects how the water particle velocity and acceleration decays exponentially from the sea surface. A zero or negative value signals an infinite water depth formulation.
- ④ **Gravitation** - Specifies the gravity vector. Its direction also defines the normal of the calm sea surface, i.e., the normal vector for the sea surface is taken as  $\mathbf{n} = -\mathbf{g}/|\mathbf{g}|$  where  $\mathbf{g}$  is the specified gravity vector.



**Note:** The gravity vector may also be defined in the Model Preferences dialog box (see [Section 4.9, "Model preferences"](#)). If the gravity vector is changed there, it will automatically be updated in the Sea Environment dialog box also, and vice versa.

- ⑤ *Wave direction* - Specifies the global wave propagation direction. This vector is used to construct the reference coordinate system that is used for the wave kinematics calculations. See [Section 6.1.1, "Coordinate system for wave kinematics"](#) below.
- ⑥ *Marine growth* - Specifies properties that are used to calculate additional mass due to marine growth on submerged structures. See [Section 6.1.2, "Marine growth"](#) below.
- ⑦ *Wave function* - Specifies a function that describes the sea surface elevation (above the specified mean sea level) and the water particle motion, as function of space and time. See [Section 6.2.1, "Sea wave functions"](#) below.
- ⑧ *Current function* - Specifies a function that describes the current magnitude (i.e., the water particle velocity), as function of depth (and time). See [Section 6.2.1, "Sea current functions"](#) below.
- ⑨ *Current direction* - Specifies the angle (in radians) between the wave propagation direction, and the current direction, as function of depth (and time). See [Section 6.2.1, "Sea current functions"](#) below.
- ⑩ *Current scale* - Specifies an optional scaling of the current magnitude. Any defined general Function can be given here.
- ⑪ *Hydrodynamic force scale* - Specifies a optional scaling of the computed hydrodynamic forces. Any defined general Function can be given here. This can be used to, e.g., ramp up the hydrodynamic forces in the beginning of a simulation with a sensitive structure.

### 6.1.1 Coordinate system for wave kinematics

The wave kinematics (water particle velocities and accelerations) are referred to a coordinate system which is defined as follows:

- Its origin is the projection of the global origin onto the MSL surface.
- Its local Z-axis coincides with the normal vector (**n**) of the MSL surface, which is defined from the gravity vector (**g**) through  $\mathbf{n} = -\mathbf{g}/|\mathbf{g}|$ .
- Its local X-axis is the projection of the specified *Wave direction* vector onto the MSL surface. However, if an RAO is used (see [Section 6.3, "Response amplitude operators"](#)), this wave direction vector is not used. Instead, the local X-axis of the *Vessel Triad* in the initial configuration is projected onto the MSL surface. This projected vector is then rotated the selected *Wave direction* angle (see bullet ⑥ in [Section 6.3.1, "Creating RAOs"](#)) about the local Z-axis to become the resulting local X-axis.
- Its local Y-axis is the cross product of the local Z- and X-axes.

### 6.1.2 Marine growth

Fixed marine structures in shallow waters are often burdened by extra weight due to marine growth over the years of operation. This added mass contribution may have a significant impact of the dynamic behavior of the structure, when subjected to loads due to wave and/or current.

Fedem offers the possibility to include marine growth effects through the following fields in the Sea Environment dialog box (shown to the right):

Marine growth			
Density	<input type="text" value="0.0"/> ①	Upper limit	<input type="text" value="0.0"/> ③
Thickness	<input type="text" value="0.0"/> ②	Lower limit	<input type="text" value="0.0"/> ④

- ① *Density* - Average mass density  $\rho_{mg}$  of the marine growth
- ② *Thickness* - Average thickness  $t_{mg}$  of the marine growth
- ③ *Upper limit* - The water depth  $z_u$  relative to the MSL, at which the marine growth starts
- ④ *Lower limit* - The water depth  $z_l$  relative to the MSL at which the marine growth ends

Added mass due to marine growth is computed for all beam elements satisfying  $z_l \leq z_c \leq z_u$ , where  $z_c$  denotes the z-coordinate of the centre of gravity of the beam, with respect to the MSL. For a beam element with length  $L$  and outer diameter  $D_o$  the marine growth mass is computed as

$$M_{mg} = \rho_{mg} \pi (D_o + t_{mg}) t_{mg} L$$

## 6.2 Wave and current functions

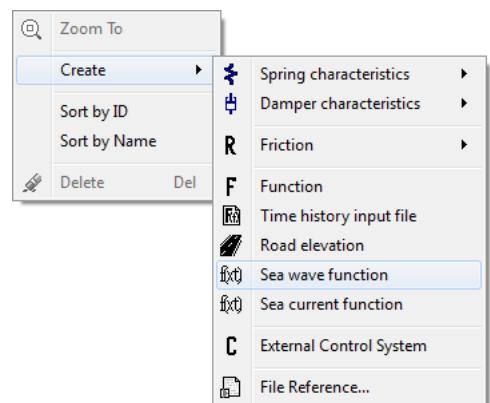


Wave and current is modeled as *Functions* in Fedem, through two specialized classes of functions, *Sea wave function* and *Sea current function*. They have many of the same properties as the general functions, described in [Section 5.11, "Functions"](#), but with some special properties in addition which are described in the following.

### 6.2.1 Sea wave functions

Sea wave functions represent the kinematic properties of sea waves. In addition to describing the elevation of the sea surface at a given point and time, they also represent the velocity and acceleration of the water particles from the sea surface down to the sea bed, which are needed in order to calculate hydrodynamic loads acting on submerged structures.

You create a wave function by right-clicking an empty space in the Model Manager *Objects* list, selecting **Create** and then **Sea wave function**. You can also access the command from the *Marine* menu. The new wave function, which is automatically selected, is added to the list of *Wave functions* maintained in the Model Manager *Objects* list.



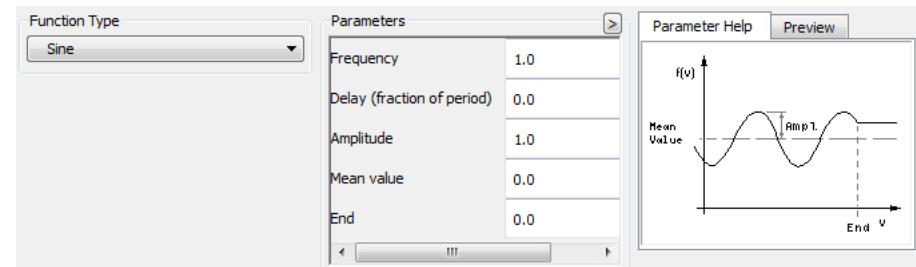
The properties of the created wave function are shown in the Property Editor panel. It is similar to the property panel of the general functions (see [Section 5.11, "Function properties"](#)), except that the *Argument* frame is absent. Wave functions are always functions of spatial location and time. The selection of *Function Type* is also limited to the following periodic function types:

- [Sine](#)
- [JONSWAP sea wave spectrum](#)
- [User defined wave spectrum](#)

### Sine

Wave functions of type *Sine* are used to model regular waves, defined from a constant amplitude,  $A$ , angular frequency,  $\omega$ , and phase shift,  $\varepsilon$ .

The property panel of the Sine wave function is shown below



**NOTE:** This property panel is the same as that of general Sine functions (see "[Sine](#)" in [Section 5.11.5](#)). Thus, the Frequency and Delay parameter values specified here are multiplied by  $2\pi$  to become the  $\omega$  and  $\varepsilon$  parameters, respectively. The Mean value and End parameters are not used in Wave functions.

In case of *infinite* water depth, the wave elevation above the mean sea water level,  $h$ , the horizontal and vertical water particle velocities,  $\{v_x, v_z\}$ , and the horizontal and vertical water particle acceleration,  $\{a_x, a_z\}$ , are given as the following functions of time,  $t$ , and the spatial coordinates,  $x$  and  $z$ , referring to the coordinate system defined in [Section 6.1.1, "Coordinate system for wave kinematics"](#):

$$h = A \sin(\omega t - kx + \varepsilon)$$

$$\begin{bmatrix} v_x \\ v_z \end{bmatrix} = \omega A e^{kz} \begin{bmatrix} \sin(\omega t - kx + \varepsilon) \\ \cos(\omega t - kx + \varepsilon) \end{bmatrix}$$

$$\begin{bmatrix} a_x \\ a_z \end{bmatrix} = \omega^2 A e^{kz} \begin{bmatrix} \cos(\omega t - kx + \varepsilon) \\ -\sin(\omega t - kx + \varepsilon) \end{bmatrix}$$

where  $k = \omega^2/g$  is the wave number ( $g = |\mathbf{g}|$  being the gravity constant).

In the case of *finite* water depth (see bullet ③ in [Section 6.1, "Sea environment"](#)), the expression for the wave elevation is the same as above, but the wave number,  $k$ , is now determined from the following nonlinear equation

$$\frac{\omega^2}{g} = k \tanh kd$$

where  $d$  is the average water depth. This equation is solved through Newton iterations before the time integration is started for each wave component. The water particle velocity and acceleration are then calculated from the following expressions during the time integration:

#### **Wheeler stretching**

The expressions above for the water particle velocity and acceleration have a term depending on the vertical coordinate  $z$  to model the exponential decay of the motion variables as we go deeper. This term has the value 1.0 at the water surface ( $z=0.0$ ) and then decreases exponentially towards zero. However, above the mean water level ( $0 < z \leq h$ ), i.e., inside a wave crest, this term will be larger than 1.0 and then give an unreasonable high velocity and acceleration. To avoid this,

$$\begin{bmatrix} v_x \\ v_z \end{bmatrix} = \frac{\omega A}{\sinh kd} \begin{bmatrix} \cosh(kz + kd) \sin(\omega t - kz + \varepsilon) \\ \sinh(kz + kd) \cos(\omega t - kz + \varepsilon) \end{bmatrix}$$

$$\begin{bmatrix} a_x \\ a_z \end{bmatrix} = \frac{\omega^2 A}{\sinh kd} \begin{bmatrix} \cosh(kz + kd) \cos(\omega t - kz + \varepsilon) \\ -\sinh(kz + kd) \sin(\omega t - kz + \varepsilon) \end{bmatrix}$$

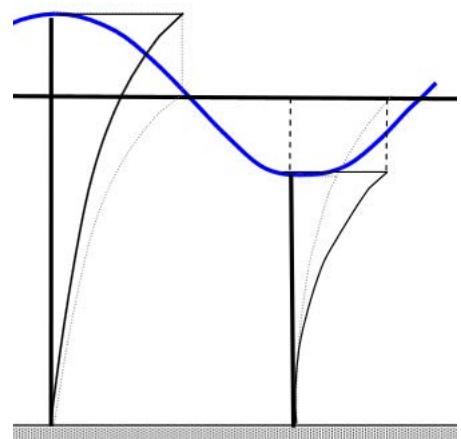
Fedem employs *Wheeler stretching* on the velocity/acceleration expressions. That is, the z-coordinate is replaced a modified value  $z'$ , where

$$z' = z - h \quad \text{for infinite water depth}$$

$$z' = \frac{z - h}{d + h} \cdot d \quad \text{for finite water depth}$$

For the case of finite water depth, the effect of this modification is illustrated in the figure to the right. As can be seen, the magnitude is reduced inside a wave crest, but increased below a wave trough.

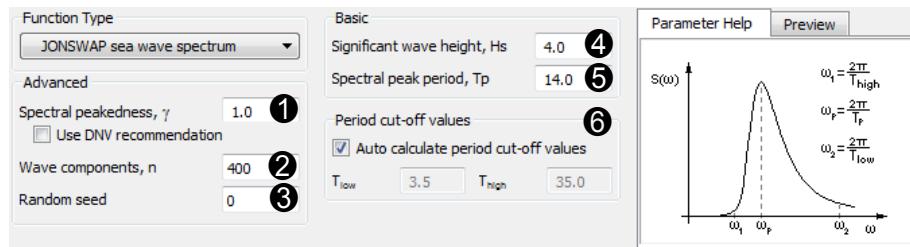
Alternatively, a simple vertical stretch may be employed, which means we use  $z' = 0.0$  when  $z > 0.0$ . To use this formulation, specify `-noWheelerStretching` in the Additional Solver Options dialog box, *Dynamics Solver* field, see [Section 8.2.3, "Additional solver options"](#), before the simulation is started.



### JONSWAP sea wave spectrum

Wave functions of type *JONSWAP (Joint North Sea Wave Project) sea wave spectrum* are used to model irregular waves. Basically, these functions consist of a sum of simple Sine wave functions, as defined above, where the Amplitude, Frequency and Phase delay parameters varies for each component. These parameters are calculated from a few user-specified parameters, such that the resulting wave function exhibits some predefined statistical properties.

The property panel of the JONSWAP sea wave function is shown below.



- ① **Spectral peakedness** - This is a non-dimensional parameter ( $\gamma$ ) that defines the peakedness of the wave components. The default value is 3.3. If the *Use DNV recommendation* toggle is enabled,  $\gamma$  is calculated automatically from the specified significant wave height and spectral peak period based on the formulas given below.
- ② **Wave components** - Specifies the number of wave components ( $n$ ).
- ③ **Random seed** - This is a non-negative integer value that is used as random seed in the pseudo-random generator, which delivers a sequence of random values in range [0.0,1.0]. These random values are used to compute the phase delay ( $\varepsilon \in [0, 2\pi]$ ) for each wave component, and the size of the frequency intervals ( $\Delta\omega_j = \omega_j - \omega_{j-1}$ ) between each wave component.
- ④ **Significant wave height** - This value specifies the significant wave height ( $H_s$ ) from trough to crest. The value is given in meters, or the current length dimension used.
- ⑤ **Spectral peak period** - This value specifies the peak period ( $T_p$ ) of the wave spectrum in seconds.
- ⑥ **Period cut-off values** - This defines the frequency range to use in the wave spectrum ( $\omega_0=2\pi/T_{high}$  and  $\omega_n=2\pi/T_{low}$ ). If the *Automatically calculate period cut-off values* toggle is enabled, the  $T_{low}$  and  $T_{high}$  values are calculated automatically based on the formulas below.

Based on the property parameters above, the amplitude,  $A_j$ , for wave component  $j$  with a given angular frequency  $\omega_j$  is computed as

$$A_j = \sqrt{2S(\omega_j)\Delta\omega}$$

where  $S(\omega)$  denotes the actual wave spectrum and  $\Delta\omega$  is a constant difference between successive frequencies in the spectrum.

The JONSWAP spectrum is defined as follows<sup>1</sup>:

$$S(\omega) = A_\gamma \cdot \frac{5}{16} \cdot \frac{H_s^2 \omega_p^4}{\omega^5} \cdot \exp\left(-\frac{5}{4}\left(\frac{\omega}{\omega_p}\right)^{-4}\right) \cdot \gamma^Y$$

$$A_\gamma = 1 - 0,287 \ln(\gamma) \quad Y = \exp\left(-\frac{1}{2}\left(\frac{\omega - \omega_p}{\sigma \omega_p}\right)^2\right)$$

where  $H_s$  is the significant wave height,  $\omega_p = 2\pi/T_p$  is the spectral peak frequency,  $\gamma$  is the peakedness parameter and  $\sigma$  is a spectral shape parameter. The latter is set equal to 0.07 for frequencies  $\omega$  less than or equal to  $\omega_p$  and to 0.09 for frequencies greater than  $\omega_p$ .

The spectral peakedness is calculated based on recommendation in the DNV-RP-C205 (see footnote 1 below), if the the *Use DNV recommendation* toggle is enabled in the property panel (see above). This reads as follows

$$\gamma = \begin{cases} 5 & \forall \tau \leq \frac{18}{5} \\ \exp\left(\frac{23}{5} - \frac{23}{20}\tau\right) & \forall \frac{18}{5} < \tau < 5 \\ 1 & \forall \tau \geq 5 \end{cases}$$

where the parameter  $\tau$  is defined as  $\tau = \frac{T_p}{\sqrt{H_s}}$ .



**NOTE:** If the peakedness parameter  $\gamma$  is set to 1.0, such that  $A_\gamma = \gamma^Y = 1.0$ , the JONSWAP wave spectrum becomes equal to the so-called Pierson-Moskowitz Spectrum.

---

1. See S. K. Chakrabarti, *Hydrodynamics of Offshore Structures*, WIT Press, 1987, pp. 105-116 and the DNV-RP-C205 *Environmental conditions and Environmental loads*, October 201, pp. 33-34 for full development of the wave spectrum formulae.

If the *Automatically calculate period cut-off values* toggle is enabled in the property panel, the highest and lowest period values are calculated based on the peak period,  $T_p$ , and the peakedness parameter,  $\gamma$ , through

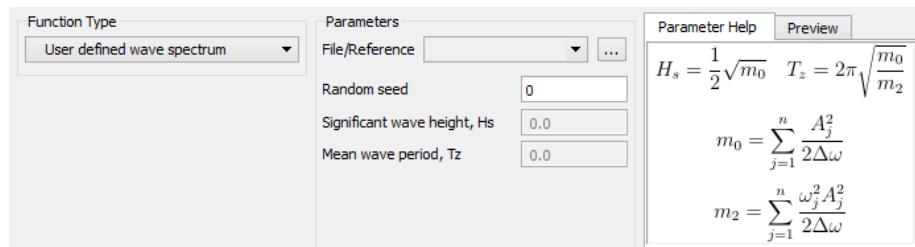
$$T_{low} = T_p \left( \frac{4\tilde{m}_0^{cut}}{5A_\gamma} \right)^{1/4}, \quad T_{high} = T_p \left( \frac{\ln(A_\gamma/\tilde{m}_0^{cut})}{5/4} \right)^{1/4}$$

where  $A_\gamma$  is as defined above, and the value  $\tilde{m}_0^{cut}$  is set to 0.0025 (0.25%). See the DNV RP-C205 for further elaboration.

### User defined wave spectrum

Wave functions of type *User defined wave spectrum* are similar to the *JONSWAP sea wave spectrum* functions, except from that the Amplitude, Frequency and Phase delay parameters (and optionally also the wave numbers) for each wave component, now are read from an ASCII file instead of calculating them from input parameters.

The property panel of the User defined wave spectrum function is shown below. It contains a *File/Reference* field through which the ASCII file containing the wave spectrum is specified.



The specified file is assumed to consist of either 2, 3 or 4 white-space separated columns of data, with the following interpretation:

1. Wave amplitude [m]
2. Frequency [Hz]
3. Phase delay in fraction of the wave period [0,1]
4. Wave number ( $k$ ) [ $m^{-1}$ ]

The number of columns must be specified on the first line in the file, via the string "#ncol <n>" where <n> is the value 2, 3 or 4. If the first line does not contain the string #ncol, a two-column file is assumed. If the file consists of either two or three columns, the wave numbers are

calculated automatically based on the gravitation constant ( $g$ ), the angular frequency ( $\omega$ ), and the water depth ( $d$ ) in case for finite depth formulation, as described above for the *Sine* wave functions. If the file consists of two columns only, the phase delay of each wave component are assigned using the internal pseudo-random generator, as for the *JONSWAP sea wave spectrum* functions.

The property panel also contains two fields displaying the Significant wave height ( $H_s$ ) and the Mean wave period ( $T_z$ ), which are derived from the provided amplitude ( $A_j$ ) and frequency data ( $\omega_j$ ) based on the formulas shown in the *Parameter Help* view tab.

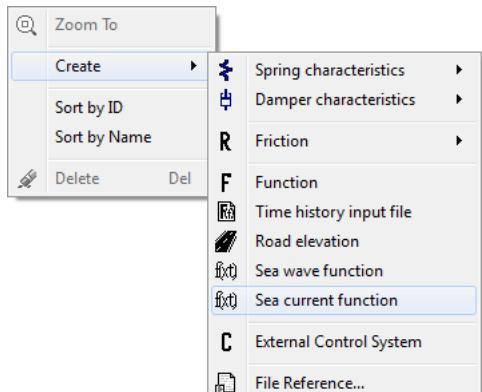
## Sea current functions

Sea current functions specify the current velocity magnitude and direction as functions of water depth and time. The resulting current velocity vector is superimposed to the velocity of the specified wave function, if any, to get the overall fluid velocity on which the hydrodynamic load calculations are based.



**NOTE:** It is assumed that the time dependency on the current velocity is so weak that the water particle acceleration derived from it can be ignored. Thus, no acceleration-dependent hydrodynamic loads will result from the specified current.

You create a current function by right-clicking an empty space in the Model Manager *Objects* list, selecting **Create** and then **Sea current function**. You can also access the command from the *Marine* menu. The new current function, which is automatically selected, is added to the list of *Current functions* maintained in the Model Manager *Objects* list.



The properties of the created current function are shown in the Property Editor panel. It is identical to the property panel of the general functions (see [Section 5.11, "Function properties"](#)), except that the *Argument* frame is absent. The current functions are functions of the vertical coordinate  $z$  of the wave coordinate system only (see [Section 6.1.1, "Coordinate system for wave kinematics"](#)). However, if the type is set to *Math Expression*, a function of all the spatial coordinates  $x, y, z$  (referring to the wave coordinate system), and the time  $t$  may be defined.

## 6.3 Response amplitude operators

The goal of a vessel Response Amplitude Operator (RAO) is to express the motion of a point on a rigid vessel floating in the sea as a function of the wave height at the same reference point. The concept of a RAOs is associated with the stationary sinusoidal response (or motion) of a linear system due to a stationary sinusoidal load (wave). Due to the assumption of system linearity, the following is true:

1. The response amplitude is proportional to the wave amplitude for a given wave period, e.g., twice the wave amplitude will give twice the response amplitude.
2. The response period is exactly equal to the wave period, e.g., a sinusoidal wave with a period of 10 seconds will cause a sinusoidal response with a period of 10 seconds.
3. The response and the wave are usually not in phase. Therefore, the extreme value of the wave and the response usually do not occur at the same time.

For a reference point at the origin of the wave coordinate system (see [Section 6.1.1, "Coordinate system for wave kinematics"](#)), the wave height for a regular wave (or for one component of an irregular wave), is written as the following function of time ( $t$ ):

$$h = A \cdot \sin(\omega t + \varepsilon)$$

The associated motion response is then assumed on the form

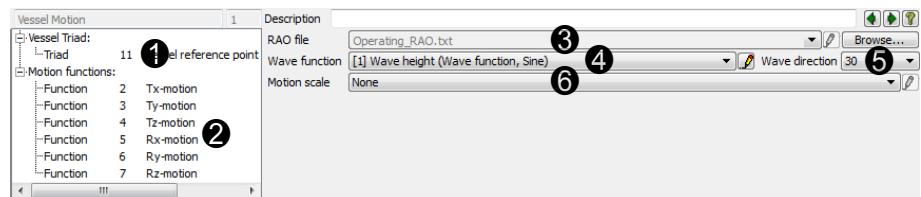
$$R = B \cdot A \cdot \sin(\omega t + \varepsilon + \phi)$$

where  $A$  is the wave amplitude,  $\omega$  is the angular frequency and  $\varepsilon$  is the phase delay of the wave, whereas  $B$  and  $\phi$  are the relative amplitude and phase angle, respectively, of the RAO. Tables of  $B$  and  $\phi$  are provided for a given vessel for each of the six DOFs in the reference point, and for different incoming wave angles, see [Section 6.3.1, "Creating RAOs"](#).

### 6.3.1 Creating RAOs



You can create an RAO in Fedem by selecting **Response Amplitude Operator** in the *Marine* menu. An object of type *Vessel Motion* is then created, along with six general functions representing the motion of the vessel in the six individual DOFs of the vessel's reference point. The created Vessel Motion object is automatically selected, such that its topology view and property panel (shown below) are displayed.



6

- ❶ **Vessel Triad** - This specifies the reference point of the vessel. When at least one of the Motion functions is assigned as prescribed motion in a given Triad, that one is automatically recognized as the Vessel Triad.
- ❷ **Motion functions** - Here, the created six functions that represent the actual motion of the vessel reference point are listed:
  - a. Tx-motion - translation in local X-direction
  - b. Ty-motion - translation in local Y-direction
  - c. Tz-motion - translation in local Z-direction
  - d. Rx-motion - rotation about local X-axis
  - e. Ry-motion - rotation about local Y-axis
  - f. Rz-motion - rotation about local Z-axis
 These functions are all on the form as given by the second equation above (see page 6-12), where the  $A$ ,  $\omega$  and  $\varepsilon$  values are taken from the selected *Wave function*, whereas  $B$  and  $\phi$  are read from the *RAO file*.
- ❸ **RAO file** - This field specifies a file containing the RAO table(s) describing the vessel motion as functions of wave elevation and frequency. Use the **Browse...** button to select the file to be used. See [Section 6.3.2, "RAO table format"](#), for description of the RAO file format.
- ❹ **Wave function** - This field specifies the wave function defining the wave height ( $A$ ), angular frequency ( $\omega$ ) and phase delay ( $\varepsilon$ ) used by the motion functions. Normally, this should be the same function as specified as the Wave function in the Sea Environment dialog box (see [Section 6.1, "Sea environment"](#)), but is not necessarily so.
- ❺ **Wave direction** - This pull-down menu contains the list of wave direction angles that are specified in the selected RAO file. They define the angle (in degrees) between the local X-axis of the *Vessel Triad* and the incoming wave direction, i.e., a zero wave direction means that the waves propagate in the direction of the negative local X-axis of the Vessel Triad and the value 90.0 means that the waves propagate in the direction of the negative Y-axis. A separate RAO table is associated with each of the direction angles in the RAO file.



**NOTE:** In addition to selecting the actual RAO table to use, the Wave direction angle also affects the wave kinematics (i.e., the water particle velocity and acceleration), and thereby also the hydrodynamic forces acting on the submerged part of the mechanism, since it is also used to define the local X-axis of the wave coordinate system, relative to the local axes of the Vessel Triad.

- ⑥ **Motion scale** - This field specifies a General Function to be used as a scaling on the six generated *Motion functions* in bullet ②. This can be used to ramp up the RAO motions in a smooth manner, to avoid undesired startup transients. When a function is specified here, six new motion functions are generated, where each one is defined as the product of the original motion function and the scaling function.

After creating the Vessel Motion object, you must manually assign the six created motion functions as Prescribed motions to the Triad that defines the vessel reference point. All six functions must be assigned to the same Triad. It is also possible to leave out some of the functions, if, for instance, some are identically zero and/or you want to study the response in 2D only. However, if some of the functions are referred by another Triad (or Joint), a *Vessel Triad* is not identified and the RAO will not work.

If a *Motion scale* function is specified after you have assigned the original motion functions to the vessel reference triad, those functions are automatically replaced by the scaled versions. Moreover, if you later change the Motion scale function back to *None*, the scaled motion functions are removed again and the references to the original functions are restored. Thus, you only need to change the *Motion scale* function setting to efficiently toggle the RAO scaling on or off.

### 6.3.2 RAO table format

A sample RAO file is shown below. An RAO file consists of a series of tables, each with 14 columns of data. Each table is started by the line

`Direction <angle>`

where `<angle>` is the wave direction as discussed above (see [Section 6.3.1, "Creating RAOs"](#)). Any line starting with an apostrophe ('') is treated as comments, and can be used for increased human readability.

```
'Excerpt from a sample RAO file
Direction 0
'Freq. Period -- Surge --- Sway --- Heave --- Roll --- Pitch --- Yaw ----
'          Ampl. Phase Ampl. Phase Ampl. Phase Ampl. Phase Ampl. Phase
'rad/s)   (s)   (m/m) (deg) (m/m) (deg) (m/m) (deg) (deg/m) (deg) (deg/m) (deg)
0.209    30.0  1.304 -90.0  0.0   -87.0  1.047  -4.0  0.0    93.0  0.064  24.0  0.0   77.0
0.233    27.0  1.189 -90.0  0.0   -4.0   1.073  -5.0  0.0   176.0  0.124  74.0  0.0   25.0
0.251    25.0  1.114 -90.0  0.0    4.0   1.108  -7.0  0.0  -176.0  0.173  81.0  0.0   33.0
...
Direction 15
0.209    30.0  1.260 -90.0  0.333 -90.0  1.047  -4.0  0.019 -95.0  0.062  25.0  0.038 -178.0
...
```

The first two columns of each table list the angular *frequency* and corresponding wave *period*, respectively, of the wave components. The period value is not used, it is listed for information only. However, it can not be omitted from the file (a dummy value of 0.0 is fine though).

The angular frequencies must be listed in increasing order within each table, and linear interpolation is used between the two closest frequencies listed, for actual wave components with frequencies not present in the table. Flat extrapolation is used for values lower than the first frequency listed in the table, or higher than the last frequency listed.

The remaining 12 columns list pairs of *amplitude* scales and *phase* delays (the  $B$  and  $\phi$  quantities as defined through the second equation on page 6-12) for the six DOFs (Surge, Sway, Heave, Roll, Pitch and Yaw) of the vessel reference point.

## 6.4. Beam string structures

6

A beam string in Fedem is an assembly of several *Beams* (beam elements) which are connected to each other in *Triads* along an (initial) straight line. A beam string is represented by a sub-assembly object of the sub-type *Beamstring* in the Fedem model (see [Section 5.15, "Sub-assemblies"](#)). It can be used to model typical one-dimensional marine structures, such as flexible risers, anchoring lines, umbilicals, etc.

### 6.4.1 Import of beam strings



You can create a beam string by selecting **Import Beamstring...** from the *Marine* menu. A plain dialog box through which you can select an input file containing the definition of the beam string and its properties then pops up. The format of the beam string definition file is described in [Section 6.4.2, "Beam string file format"](#) below.

The triads and beam elements of the beam string assembly are generated from a specified starting point, and in the opposite direction of the gravitation vector as defined in the Sea Environment dialog box (see [Section 6.1, "Sea environment"](#)). Therefore, if a non-vertical beam string is desired, you have to temporarily change the gravitation vector while performing the beam string import.

### 6.4.2 Beam string file format

The beam string input file lists the properties of the beam structure, from the starting (bottom) point and up. All blank lines and text lines in the file are ignored (treated as comments) except for a limited set of keywords and part names, as described below. The position of new-line characters is arbitrary within a data section of the file. Thus, you may add or remove as many new-line characters you wish to improve readability.

The first three numerical values in the file are assumed to be the global coordinates ( $X$ ,  $Y$ ,  $Z$ ) of the beginning of the beam. Then, an arbitrary number of the following data sections are assumed (in arbitrary order):

```

<part name>
<Rho> <E> <G> <Do> <Di> <Dd> <Db> <Cd> <Cm> <Ca>
<L> <nel>
"<vis-file>"

Ball Joint

Free Joint
<Zpos>

Cylindric Joint
<Zpos>

```

The interpretation of these keywords and values are as follows:

- <part name> - Description to be assigned to the beams (optional). Whether a part name is specified or not is determined based on the first non-space character in this line; if it is neither a digit nor any of the characters '.', '-' or '+' it is assumed that a part name is given.
- <Rho> - Structural mass density.
- <E> - Young's modulus of elasticity.
- <G> - Shear modulus of elasticity.
- <Do> - Outer pipe diameter.
- <Di> - Inner pipe diameter.
- <Dd> - Drag diameter.
- <Db> - Buoyancy diameter.
- <Cd> - Drag coefficient.
- <Cm> - Added mass coefficient for fluid.
- <Ca> - Added mass coefficient for structure.

- <L> - Total length of this beam segment.
- <nels> - The number of beam elements this segment is divided into.
- <vis-file> - Path to a visualization file describing the geometry to be displayed for each element of this beam segment (optional). Unless an absolute path is given, the path is assumed to be relative to the folder of the beam string input file.
- <Zpos> - Global position from starting point in beam string direction.
- **Ball joint** - Creates a Ball Joint at current location.
- **Free Joint** - Creates a Free Joint with master triad at current location and slave triad at the given position.
- **Cylindric Joint** - Creates a Cylindric Joint with masters at the two latest generated triads, and slave triad at the given position.

A sample beam string input file is shown below:

```

Bottom position
0.0 2.382 0.0

SBOP
Rho      E       G       Do      Di      Dd      Db      Cd      Cm      Ca      L      nels
26992.0 2.06E11 7.92E10 2.533   2.0     2.091   0.87    1.0     1.7     1.0     3.0    2
riser2
26992.0 2.06E11 7.92E10 2.433   2.0     2.091   0.87    1.0     1.7     1.0     3.0    2
riser3
26992.0 2.06E11 7.92E10 2.333   2.0     2.091   0.87    1.0     1.7     1.0     3.0    2
riser4
26992.0 2.06E11 7.92E10 2.233   2.0     2.091   0.87    1.0     1.7     1.0     3.0    2
26992.0 2.06E11 7.92E10 2.133   2.0     2.091   0.87    1.0     1.7     1.0     3.0    2
26992.0 2.06E11 7.92E10 2.033   2.0     2.091   0.87    1.0     1.7     1.0     3.0    2
26992.0 2.06E11 7.92E10 1.933   1.9     2.0     0.87    1.0     1.7     1.0     3.0    2

Free Joint

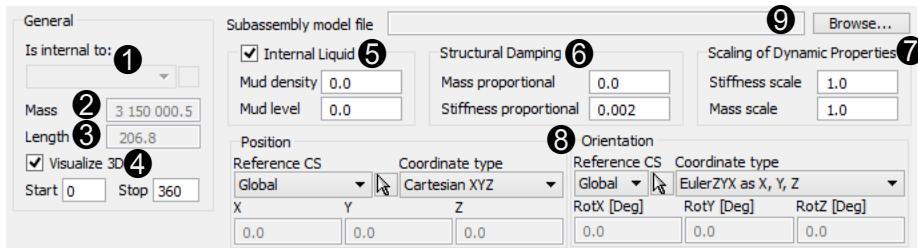
```

Each beam segment is divided into the specified number of elements of uniform length <L>/<nels>. A triad is created between each beam element that a beam segment consists of. The end triad of one segment automatically becomes the starting point of the next, or the master triad of the subsequent Joint, if specified.

A beam cross section object (see [Section 5.4, "Beam cross sections"](#)) is generated for each beam segment and assigned the structural- and hydrodynamic properties listed above (assuming a *Pipe* cross section). This object is referred to by each beam element of that segment such that hydrodynamic loads (added mass and drag forces, as well as buoyancy) can be calculated for the beam string based on these properties (see [Section 6.7.2, "Hydrodynamic loads on beam elements"](#)).

### 6.4.3 Beam string properties

The property panel of a *Beamstring* object is shown below:



- ① *Is internal to:* - Currently not used.
- ② *Mass* - This field displays the total weight of the beam string structure. It is computed as the sum of the structural mass of the individual beam elements that the beam string consists of. Additional mass due to marine growth (see [Section 6.1.2, "Marine growth"](#)) and/or internal liquid (see "[Mass from internal fluid](#)" in [Section 6.7.2](#)) is *not* included.



**TIP:** To get a mass distribution overview when the effects of the marine growth and/or internal liquid is accounted for, see the mass summary in the `fedem_solver.res` file after the Dynamics solver has been started.

- ③ *Length* - This field displays the total length of the beam string, i.e., the sum of the length of each element that the beam string consists of.
- ④ *Visualize 3D* - This toggle turns on and off the 3D-visualization of all beam elements contained in the beam string assembly. The *Start* and *Stop* fields are used to specify a sector of partial 3D visualization. They define the start and stop angles (in degrees), relative to the local Y-axis of the cross section, of the sector that is to be rendered in 3D.



**NOTE:** This Visualize 3D toggle has a tri-state value. In addition to the usual on  and off  states, it can also have a neither-on-nor-off  state, in which the corresponding on or off settings on each element are applied instead. This is usually the default setting for such tri-state toggles and indicates that the corresponding setting on the element level is not unique.

- ⑤ *Internal Liquid* - This toggle enables the calculation of additional mass due to internal liquid (mud) in the beam string (see "[Mass from internal fluid](#)" in [Section 6.7.2](#)).
  - *Mud density* - Mass density of the internal liquid.
  - *Mud level* - Global Z-coordinate of the mud level. Only those beam elements having their centre of gravity below this level is assumed to be filled with mud.

- ⑥ **Structural Damping** - Use these fields to apply common mass- and stiffness-proportional damping parameters to all beams of the beam string.
- ⑦ **Scaling of Dynamic Properties** - Use these fields to apply common stiffness and mass scaling to all beams in the beam string.



**NOTE:** If any of the structural damping and stiffness/mass scaling fields are blank, it means that the beam elements of the beam string do not have a common value for that property. This typically occurs after you edit the Beam property directly.

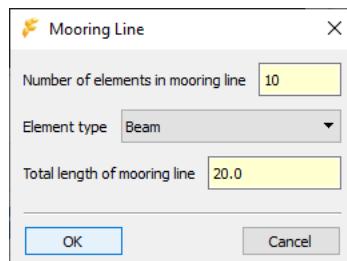
- ⑧ **Position and Orientation** - These fields can be used to adjust the global position of the beam string assembly after it has been imported. They work as those of the *Origin* property tab, see [Section 4.5.4, "Origin property"](#). An assembly can be moved freely until it is attached to ground, or connected to other mechanism objects.
- ⑨ **Subassembly model file** - A separate model file may be assigned to the beam string assembly by using the **Browse...** button. In that case, all the beams, triads and properties of this beam string are stored in the specified file when the model is saved. It is then possible to reuse this beam string in another model by importing it into an existing model (see [Section 5.15.3, "Importing sub-assemblies"](#)). If the *Subassembly model file* field is empty, the elements of the beam string are stored in the main model file together with the rest of the model.

#### 6.4.4 Mooring line generation

A mooring line is a 1D structure with very low bending stiffness that can be modeled as a chain of beam elements, and where the shape between the two end points are mostly determined by the own weight of the structure. Fedem offers a tool for easy generation of such structures in a equilibrated modeling configuration.



To create a mooring line, select two Triads in the *Modeler* view or from the *Objects* list in the Model Manager panel, and select **Mooring Line...** from the *Marine* menu. The dialog box shown below then appears.



Here, you can specify the number of elements and the total length of the mooring line to be generated. You may also change the element type to *Generic Part* instead of *Beam*, or to any other two-noded user-defined element you may have in your installation. Then select **OK** to generate the elements. If the two selected Triads are member of a common sub-assembly, the generated elements and Triads will become members of the same sub-assembly as well.

## 6.5. Space frame structures

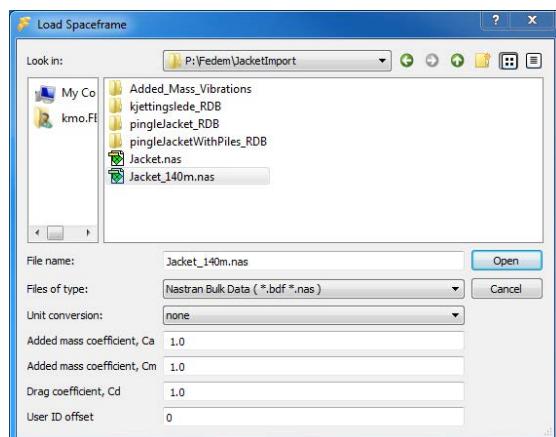
A space frame in Fedem is an assembly of several *Beams* which are connected to each other in *Triads*. An arbitrary number of beams may be connected at each Triad. Thus, it becomes a full FE representation of the structure on the system level. A space frame is represented by a Sub-assembly object of the sub-type *Jacket* in the Fedem model (see [Section 5.15, "Sub-assemblies"](#)). A Jacket is a model of a bottom-fixed off-shore structure, which are typically used as foundation for wind turbines and oil exploration platforms. The beam members of a Jacket structure may therefore be assigned properties for hydrodynamic load calculation (added mass and drag coefficients, etc., see [Section 6.7.2, "Hydrodynamic loads on beam elements"](#)).

### 6.5.1 Import of space frames



You can import a space frame by selecting **Import Spaceframe...** from the *Marine* menu. The dialog box shown to the right then pops up.

It has similar fields for selecting an FE model file as the Load Part dialog box (see [Section 5.1, "Creating parts by file import"](#)). Notice that only beam, spring and point mass elements will be read by this import. Any existing shell and solid elements present in the imported FE model file will be ignored. A complete list of the ignored elements (if any) are provided in the Output List view (see [Section 2.4.8, "Output List"](#)).

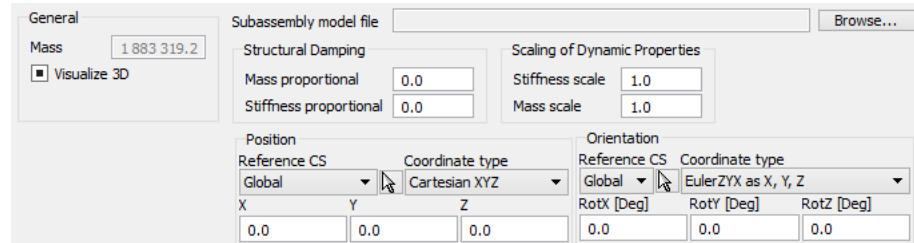


The Load Spaceframe dialog box also has fields for specification of added mass and drag coefficients ( $C_a$ ,  $C_m$  and  $C_d$ ). These will be the default values that are applied to all beam members of the imported space frame.

By default, the Beam and Triad objects that are created during the import will receive as user ID the external ID stored in the FE model file for the corresponding beam element and FE node. However, a constant offset value may be specified in the *User ID offset* field in the dialog box.

## 6.5.2 Space frame properties

The property panel of a *Jacket* or space frame object is shown below:



It is similar to the property panel of the *Beamstring* object (see [Section 6.4.3, "Beam string properties"](#)), except that the *Is internal to:* and the *Length* fields are absent (irrelevant for space frame structures), as well as the *Start* and *Stop* angle fields for 3D visualization. The *Internal Liquid* fields are also absent although the beam elements of a space frame may be liquid-filled. Instead, the mass density of the internal liquid and the liquid level are assumed identical to the *Water density* and the *Mean sea level*, respectively, as defined in the Sea Environment dialog box (see [Section 6.1, "Sea environment"](#)). Whether a beam is to be considered liquid-filled or not is in this case toggled by specifying a non-zero diameter of the internal liquid ( $D_i$ ) in the cross section referred by the beam (see "*Hydrodynamic properties*" in [Section 6.7.2](#)). See also "*Mass from internal fluid*" in [Section 6.7.2](#) for more details on calculation of additional mass from internal liquid.

The data fields present in the *Jacket* property panel has a similar interpretation as the corresponding fields found in the *beam string* property panel. We therefore refer to [Section 6.4.3, "Beam string properties"](#) for a detailed description of these fields.

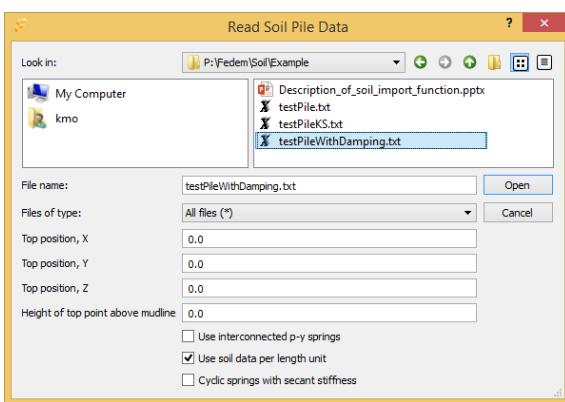
## 6.6. Soil Pile structures

A soil pile in Fedem is similar to a beam string (see [Section 6.4, "Beam string structures"](#)) in the sense that it consists of several *Beams* connected to each other in *Riads* along a straight line. In addition, the soil pile contains several *Free joints* with nonlinear spring characteristics that model the soil interaction for the pile (see figure on page 6-25). A soil pile is represented by a Sub-assembly object of the sub-type *Soil Pile* in the Fedem model (see [Section 5.15, "Sub-assemblies"](#)).

### 6.6.1 Import of soil piles



You can import a soil pile by selecting **Import Soil Pile...** from the *Marine* menu. The dialog box shown to the right then pops up, in which you can select the file that contains the soil pile definition. See [Section 6.6.2, "Soil pile file format"](#) for a description of the soil pile definition file format.



The Read Soil Pile Data dialog box has three fields (*Top position, X*, *Top position, Y* and *Top position, Z*) for specification of the global position of the top of the pile. The beams representing the pile are then generated from this top point, and in the direction of the gravitation vector.

The *Height of top point above mudline* field (HT) defines where the mudline is located, relative to the top point (see figure on page 6-25). The location of the soil springs along the pile are then specified relative to this mudline level.

The Read Soil Pile Data dialog box has in addition three toggles that affect the generated Free Joint and spring objects of the pile:

- *Use interconnected p-y springs* - Enables that the soil springs work in the radial direction with respect to the pile axis. This is achieved by setting the *Translation* spring inter-connectivity to *Cylindrical Z* for the generated Free joints, see "[Advanced joint properties](#)" in [Section 5.6.2](#).

- *Use soil data per length unit* - If enabled, the soil spring data is treated as stiffness/force per length unit and is scaled by the segment length of each beam to obtain the effective Force/Stiffness to be applied in the discrete Free joints, see [Section 6.6.2, "Soil pile file format"](#) below. This is the default setting. If you have discrete soil pile data instead, remember to switch off this toggle.
- *Cyclic springs with secant stiffness* - Enables a secant stiffness behavior when the soil is unloaded, see [Section 6.6.3, "Secant stiffness on unloading"](#) below.

## 6.6.2 Soil pile file format

The soil pile input file lists the properties of the pile, from the starting point at (or above) mudline and downwards. All blank lines and text in the file are ignored (treated as comments) except for a limited set of keywords as described in the following.

A sample input file is shown below to the right. It consists of the following sections which may be repeated arbitrarily, except for that the sections **q-z** and **dq-z** can only appear once and at the end of the file.

```

Pipe <rho> <E> <G> <OD> <ID>.          # FEDEM soil pile definition file (sample).
Depth <z> [<nel>]                         #####
sd_type                                     #
[Symmetric]                                #
x0 f0                                         #
x1 f1                                         #
...                                           #
• Pipe - Specifies the structural properties of the pile (mass density, Young's and shear moduli, outer and inner pipe diameter) valid from current Dept level until the next Pipe keyword is encountered.

• Depth - Specifies the depth below mudline (Z) of the next soil spring, and optionally the number of beam elements (nel) from the previous soil spring, or start of the pile.

• sd_type - Heading defining the soil spring/damper type. The following eight keywords are recognized:
    — p-y - Normal spring
    — dp-y - Normal damper
    — t-z - Tangential spring
    — dt-z - Tangential damper
    — r-z - Rotational spring (about pile axis)
    — dr-z - Rotational damper (about pile axis)
    — q-z - Axial spring at the bottom end of the pile
    — dq-z - Axial damper at the bottom end of the pile

• Symmetric - Optional keyword indicating that the subsequent spring/damper properties are symmetric for positive and negative deflections/velocities, such that only the force values for positive deflection/velocity need to be listed.

# Pipe      rho      E      G      OD      ID
# Depth 5.0 4
p-y      Symmetric (optional switch)
displ   line force
3.022E-02 1.347E-03
5.216E-02 2.694E-03
6.467E-02 4.041E-03
7.083E-02 5.388E-03

dp-y     Symmetric (optional switch)
velocity line force
0.1      1.0
0.5      2.0
1.0      2.5

t-z displ  line force
Symmetric
0.1      1.0

dt-z displ  line force
Symmetric
0.13022E-02 1.347E-03
5.216E-02 2.694E-03

Depth 10
p-y displ  line force
3.022E-02 1.347E-03
5.216E-02 2.694E-03
6.467E-02 4.041E-03
7.083E-02 5.388E-03

t-z displ  line force
3.022E-02 1.347E-03
5.216E-02 2.694E-03
6.467E-02 4.041E-03
7.083E-02 5.388E-03

Depth 12
q-z displ  force
3.022E-02 1.347E-03
5.216E-02 2.694E-03
6.467E-02 4.041E-03
7.083E-02 5.388E-03

dq-z velocity force
0.5      1.234

```

- $x_i f_i$  - Points on the force-deflection curve for the nonlinear soil spring (or force-velocity curve for dampers). The points must be listed for increasing values of deflection/velocity ( $x_i$ ).

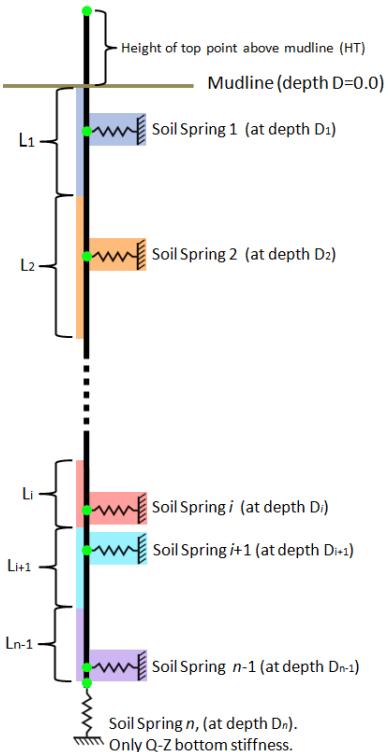
The force values ( $f_i$ ) are given as line force intensity along the pile [N/m]. The total force for a certain deflection in a spring at depth level  $D_j$  is thus  $F_j = f_j L_j$ , for  $j = 1 \dots n-1$ , where (see figure to the right):

$$L_1 = \frac{1}{2}(D_1 + D_2)$$

$$L_j = \frac{1}{2}(D_{j+1} + D_{j-1}), 1 < j < n-1$$

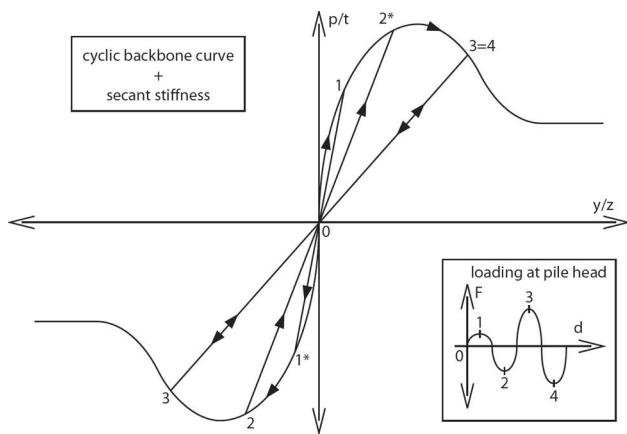
$$L_{n-1} = D_n - \frac{1}{2}(D_{n-1} + D_{n-2})$$

The bottom soil spring/damper is not subjected to this length scaling. The force values ( $f_i$ ) for these springs are therefore given in directly Newtons [N].



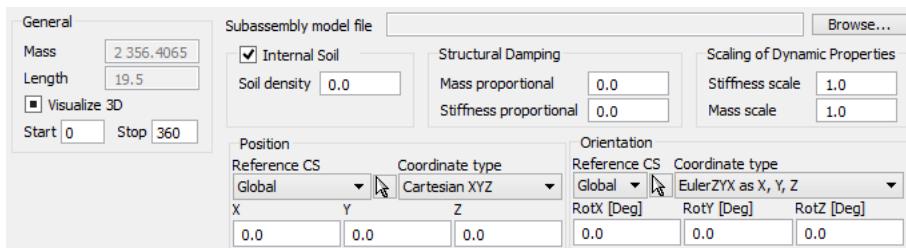
### 6.6.3 Secant stiffness on unloading

The soil springs are normally regarded as hyperelastic, i.e., when a spring is unloaded (the deflection decreases) the force follows the same force-displacement curve as when the deflection increases (and similar for the damping force versus velocity). However, when the *Cyclic springs with secant stiffness* toggle is enabled in the Read Soil Pile Data dialog box (see [Section 6.6.1, "Import of soil piles"](#)), the normal springs (p-y curves) are interpreted as cyclic back-bone curves with a secant stiffness (see figure below). The resulting secant stiffness is a function of the largest deflection experienced so far during the loading history.



#### 6.6.4 Soil pile property

The property panel of a *Soil pile* object is shown below:



It is similar to the property panel of the *Beamstring* object (see [Section 6.4.3, "Beam string properties"](#)), except that the *Is internal to:* field is absent (irrelevant for soil pile structures). The *Internal Liquid* toggle and fields are replaced by a similar toggle for *Internal Soil* and a *Soil density* field. If internal soil is enabled, the entire soil pile is assumed filled with soil and it is therefore no soil level field in this property panel. The effect of the internal soil is similar as the internal liquid in beam strings and jacket members, see "[Mass from internal fluid](#)" in [Section 6.7.2](#).

The other data fields present in the *Soil Pile* property panel has a similar interpretation as the corresponding fields found in the *Beamstring* property panel. We therefore refer to [Section 6.4.3, "Beam string properties"](#) for a detailed description of these fields.



**NOTE:** *Soil piles are also subjected to Buoyancy based on the specified water density, if they are located under the mean sea level as defined in the Sea Environment dialog (see [Section 6.1, "Sea environment"](#) and "[Hydrodynamic properties](#)" in [Section 6.7.2](#)).*

## 6.7 Hydrodynamic load calculations

### 6.7.1 Buoyancy of 3D volumes

Buoyancy forces (and associated load correction stiffness) may be included for any *Generic Part*, if the part is assigned a geometry description file in the *Visualization* field in the part property (see "[Part tab](#)" in [Section 5.1.5](#)), and the *Perform buoyancy calculations* toggle in the [Hydrodynamics tab](#) is enabled. The geometry description file has to define a closed volume that represents the total displaced fluid body when the part is submerged, and can be either on the VRML-format or the Fedem Technology Cad format ([.ftc](#), see [Section 2.2.2, "FTC format"](#)).

The buoyancy force is computed from that part of the volume that is below the specified sea water level surface and is applied in the opposite direction of the gravitation vector. In the buoyancy calculation, the water surface is assumed always as plane that contains the point

$$-\frac{s_0 \hat{g}}{\|g\|}$$

6

where  $s_0$  denotes the current sea water level, and  $\hat{g}$  is the gravitation vector. The normal vector of the sea water plane is equal to the opposite of the gravitation vector. The sea water level is either a constant (given by the *Mean sea level* parameter in the Sea Environment dialog box, see [Section 6.1, "Sea environment"](#)) or a Function of space and time, when a *Wave function* is specified in the Sea Environment dialog box. In the latter case, the current sea level is taken as the wave elevation evaluated at the local origin of the Generic Part in question.



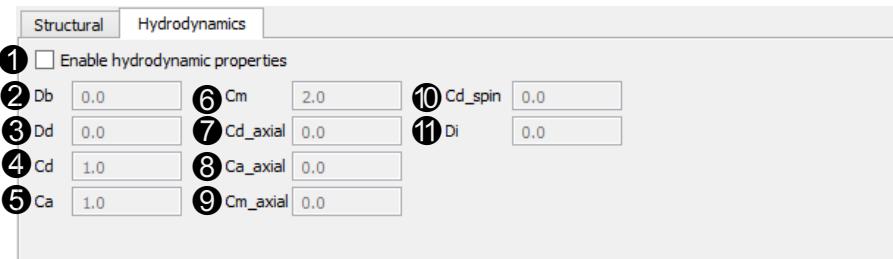
**NOTE:** When using a wave function, it is assumed that the cross section between the buoyant volume and the water surface is sufficiently small compared with the dominant wave length, as the change in water surface normal is not accounted for.

### 6.7.2 Hydrodynamic loads on beam elements

Beam elements in Fedem, that is, the basic elements of *Beamstring*, *Jacket* and *Soil Pile* components, may be subjected to hydrodynamic loads, based on the defined wave kinematics as described in [Section 6.2, "Wave and current functions"](#) and some additional hydrodynamic property parameters, see "[Hydrodynamic properties](#)" below. This provides a simplified modeling of the actual loads acting on beam structures partly or fully submerged in sea water. All hydrodynamic load calculations are performed assuming a circular outer cross section shape of the beams.

## Hydrodynamic properties

The property panel of the *Beam cross section* objects is described in [Section 5.4, "Beam cross sections"](#). In addition to the *Structural* parameters discussed there, it has a set of hydrodynamic property parameters grouped in a separate tab, as shown below:



- ① *Enable hydrodynamic properties* - Hydrodynamic loads are only calculated for beams referring to cross sections for which this toggle enabled, and that are at least partly below the defined sea surface.
- ②  $D_b$  - Effective buoyancy diameter, see "[Buoyancy of beams](#)" below
- ③  $D_d$  - Effective drag diameter, see "[Drag and added mass](#)" below
- ④  $C_d$  - Drag coefficient (in normal direction of the beam)
- ⑤  $C_a$  - Added mass coefficient associated with structure motion
- ⑥  $C_m$  - Added mass coefficient associated with water particle motion
- ⑦  $C_{d,axial}$  - Drag coefficient in axial direction of the beam
- ⑧  $C_{a,axial}$  - Added mass coefficient associated with structure motion in axial direction of the beam
- ⑨  $C_{a,axial}$  - Added mass coefficient associated with water particle motion in axial direction of the beam
- ⑩  $C_{d,spin}$  - Drag coefficient for relative torsional motion
- ⑪  $D_i$  - Diameter of internal fluid (or soil) body, used for calculation of additional mass, see "[Mass from internal fluid](#)" below



**CAUTION:** The  $D_b$  and  $D_i$  parameters in the Hydrodynamics tab are also used for beam elements in soil piles in order to compute buoyancy and mass of internal soil, if enabled (see [Section 6.6.4, "Soil pile property"](#)). All the other fields in this tab are ignored for such elements.



**NOTE:** Buoyancy forces are also calculated for beam elements in soil pile objects, based on the specified sea water density, and not the soil density. If such buoyancy is not desired, remember to set the buoyancy diameter ( $D_b$ ) to zero, or adjust the structural mass density of the beam cross section object used. Buoyancy (and mass of internal soil) is included only if *Enable hydrodynamic properties* is checked.

### Buoyancy of beams

The buoyancy force on a beam element is computed as

$$F_b = \rho_w \cdot g \cdot \pi \cdot \frac{D_b^2}{4} \cdot L$$

where  $\rho_w$  is the specified mass density of the sea water (see [Section 6.1, "Sea environment"](#)) and  $L$  is the length of the submerged part of the beam element. This buoyancy force is distributed equally on the two triads connected to the beam, but for a partly submerged beam the triad that is below the water surface receives a larger portion, and the total force is then reduced depending on how much of the beam is above the surface.

### Mass from internal fluid

When the  $D_i$  field in the *Hydrodynamics* tab is non-zero (see bullet 11 in ["Hydrodynamic properties"](#) above), additional mass is assigned to the beam elements using this cross section property. If the value is positive, the additional mass is included both as inertia forces and as a static gravity load. If it is negative, only inertia forces are included.

In either case, the additional mass is included by modifying the structural mass density as follows:

$$\rho'_s = \rho_s + \rho_i \frac{\pi D_i^2}{4A_s}$$

where  $\rho_s$  and  $\rho_i$  are the mass densities of the structural material and the internal liquid, respectively, and  $A_s$  is the net structural cross section area of the beam.

### Drag and added mass

Forces due to drag and added mass on a beam element are in Fedem computed based on the Morison equation<sup>1</sup>. This is a simplified load model valid only for slender beams with circular outer cross sections. For a completely submerged beam element, the drag and added mass forces are given as, respectively

$$F_d = \rho_w \cdot \frac{D_d}{2} \cdot L \cdot C_d \cdot (u - u_w) |u - u_w|$$

$$F_a = \rho_w \cdot \pi \cdot \frac{D_d^2}{4} \cdot L \cdot (C_a \dot{u} - C_m \dot{u}_w)$$

where  $u$  and  $u_w$  denote the structure and water particle velocities, respectively. The forces are calculated in the two normal directions of the beam element, and also in the tangential direction if axial drag and added mass coefficients are provided.



**NOTE:** For beam elements subjected to marine growth (see [Section 6.1.2, "Marine growth"](#)), the user-defined drag diameter  $D_d$  is modified to account for this, i.e. the actual value used in the above equations is  $(D_d + 2t_{mg})$ , where  $t_{mg}$  is the specified thickness of the marine growth layer. Thus, marine growth results in increased drag and added mass, in addition to the extra static weight.

The drag and added mass forces are computed individually at each end point since they also depend on the current structure- and fluid motion at that point. For a partly submerged beam element, the length  $L$  is reduced according to how much is below the sea surface. The force at the surface intersection point is in that case eccentricity-transformed onto the end that is above the water. Mass- and damping matrix contributions are also computed, based on a differentiation of the above equations.

---

1. See, e.g., S. K. Chakrabarti, *Hydrodynamics of Offshore Structures*, WIT Press, 1987, pp. 169-194 for an elaboration.

# Chapter 7 Control System Modeling

Real mechanisms are often connected to or acted upon by control elements such as sensors, controllers, and actuators. A control system is therefore necessary to simulate these effects for Fedem mechanisms. This chapter describes the control elements available in the Fedem Control System library, and explains how to model your control systems.

Sections in this chapter address the following topics:

- [Control modeling environment](#)
- [Input and output](#)
- [Control blocks](#)
- [Building control modules](#)

## 7.1 Control modeling environment

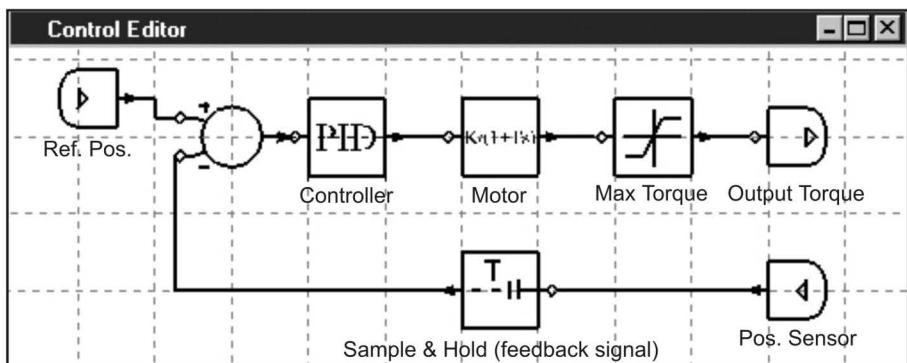
The block-diagram presentation of the control systems in Fedem closely resembles what is given in most textbooks about basic control theory. The graphical representation consists of a series of connected control blocks, which you can model to simulate your control requirements.

### 7.1.1 Control Editor

To create a control system, Fedem provides the *Control Editor* view in which control systems can be created and manipulated. Control blocks are selected from the *Control* menu or the *Control Creation* tool bar for placement in the *Control Editor* view. They can then be moved and manipulated with drag-and-drop functionality. This editing environment also features grid and snap functionality (see [Section 7.4.1, "Setting Grid and Snap"](#)).



To open the Control Editor, click the **Show Control Editor** button on the Windows menu or tool bar. The *Control Editor* view is shown below with an example control system.



### 7.1.2 Control tool bars

Fedem provides two tool bars for modeling control systems:

#### Control Creation tool bar

The *Control Creation* tool bar (shown below) consists of the Fedem control blocks that are used to build control modules. (See [Section 7.3, "Control blocks"](#) for descriptions of each control block.)



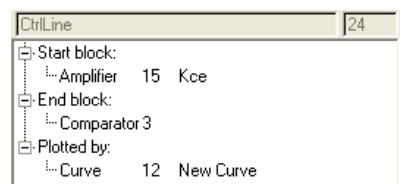
### Control Tools tool bar

The *Control Tools* tool bar (shown below) consists of drawing and editing tools. (See [Section 7.4, "Building control modules"](#) for use of these commands.)



#### 7.1.3 Control system topology

In a similar manner as for the structural mechanism, the topology of the current control system may be browsed in the *Topology* view in the lower left corner of the Fedem main window. (See also [Section 2.4.4, "ID and Topology panel"](#).)



## 7.2 Input and output

The input and output blocks are the connections through which the control system and the rest of the mechanism model interact.

7

### Control Input



The input block is used to set up input values from the mechanism, either measurements, or functions of time. The options of the Control Input element is nearly identical to the options of a Function. See [Section 5.11.2, "Function properties"](#). The output value of the Control Input is then connected to other control elements.

### Control Output



The output block is used to make a response variable from the control system available to the mechanism. When created, it will automatically be available in the various drop-down menus that allow values from a control system. The variable can then be used to control the magnitude of loads, changes in spring length, and so on.

The Control output has the option to use an embedded function to alter the output from the control system before it is applied to the mechanism. This function is edited in the same way as Functions. See [Section 5.11.2, "Function properties"](#).

## 7.3 Control blocks

*Control blocks* (also called control elements) calculate the output as a function of one or more inputs and the block's internal state. When a control element is selected in the *Control Editor* view, its mathematical equation and properties are shown in the *Property Editor* panel (see [Section 7.4.4, "Editing block properties"](#)). For more information about individual control elements, see the Fedem 7.6 Theory Guide, *Chapter 8, "Control System"*.

The following sections describe the control elements available for use in Fedem control systems.

### 7.3.1 Amplifiers

The control system supports the two types of amplifiers described below:



#### Amplifier block

The Amplifier block amplifies the input signal with a user-defined value.



#### Power block

The power block calculates and outputs the power ( $n$ ) of the input signal ( $x$ ), where  $n$  is specified by the user.

### 7.3.2 Binary-input blocks

Binary-input control elements have two input signals. The binary input blocks used in Fedem have no editable parameters. Each block is described below. (See also the Fedem 7.6 Theory Guide, *Section 8.4.1, "Basic elements."*)



#### Comparator block

The comparator block calculates and outputs the difference between the two input signals.



#### Adder block

The Adder block adds the two input signals and outputs the sum.



#### Multiplier block

The multiplier block multiplies the two input signals and outputs the product.

### 7.3.3 Integrator and limited derivator blocks

Fedem supports the use of the integrator and limited derivator blocks described below. (See also the Fedem 7.6 Theory Guide, *Section 8.4.1, "Basic elements"*.)



#### Integrator block

The Integrator block outputs the integral of the input signal.



#### Derivator block

The Derivator block outputs the derivative of the input signal within a specified bandwidth.

### 7.3.4 Time-dependent blocks

The control system supports two time-dependent control blocks described below. (See also the Fedem 7.6 Theory Guide, *Section 8.4.2, "Time-dependent elements"*.)



#### Delay block

The Delay block outputs a delayed input signal.



#### Sample-and-Hold block

The Sample-and-Hold block outputs a sampled input signal.

### 7.3.5 Non-continuous blocks

Non-continuous control blocks are used for simulating nonlinear behavior. They are used when the system response cannot be described by a linear model. (See also the Fedem 7.6 Theory Guide, *Section 8.4.3, "Piecewise continuous elements"*.)



#### Logical-Switch block

The Logical-Switch block returns one of two predefined constant inputs, both of which are dependent upon the value of a third input, called the control input. If the signal of the control input is greater than or equal to the threshold parameter, the block returns the user-specified upper limit; otherwise, it returns the user-specified lower limit.



### Limiter block

The Limiter block imposes upper and lower bounds on a signal. When the input signal is within the user-specified range of the upper and lower parameters, the input signal passes through unchanged. When the input signal is outside these limits, the signal is limited to the upper or lower limit.



### Dead-Zone block

The Dead-Zone block generates zero output within a specified range called its *dead zone*. The lower and upper limits of the dead zone are specified as the *Left* and *Right* parameters in the Property Editor panel. The block output depends on the input and the dead zone as follows:

- If the input is within the dead zone (greater than the lower limit and less than the upper limit), the output is zero.
- If the input is greater than or equal to the upper limit, the output is the input minus the upper limit.
- If the input is less than or equal to the lower limit, the output is the input minus the lower limit.



### Hysteresis block

The Hysteresis (backlash) block controls output in such a way that a change in input causes an equal change in the output. However, changes in direction of the input signal have no effect on the output. The amount of side-to-side play in the system is referred to as the *deadband*. The deadband is centered about the output.

## 7.3.6 PI, PD, and PID controllers

PI, PD, and PID controllers are used to control output in such a way that the given input source forces a desired result. PI, PD, and PID controllers are described in detail in the Fedem 7.6 Theory Guide, *Section 8.4.4, "Compensator elements"*. The following are controllers used in Fedem:



- PID Controller block
- PI Controller block
- PD Controller block
- P Limited I block (limited PI controller)



- P Limited D block (limited PD controller)
- PI Limited D block (limited PID controller in serial form)
- P Limited I and D block (limited controller in serial form)

### 7.3.7 General-transfer functions

General transfer functions are continuous mathematical functions used to describe differential equations. The time response of the system is characterized by poles (the roots of the functions). Fedem uses the general transfer functions described in the following paragraphs. For a detailed description, see the Fedem 7.6 Theory Guide, *Section 8.4.5, "General transfer functions"*.



#### Real-Pole block

The Real-Pole block is a first-order transfer function. The Real-Pole block has a pole positioned on the real, negative axis in the *s*-plane.



#### Complex Conjugate Pole block

The Complex Conjugate Pole block is described by a second-order differential equation.



#### First-Order Transfer Function block

The First-Order Transfer Function block is described in detail in the Fedem 7.6 Theory Guide (see "*First-order element*" in *Section 8.4.5*).



#### Second-Order Transfer Function block

The Second-Order Transfer Function block is described in detail in the Fedem 7.6 Theory Guide (see "*Second-order element*" in *Section 8.4.5*).

## 7.4 Building control modules

The control module is fully defined when all blocks are connected and the inputs and outputs are attached to sensors and actuators in the mechanism model.

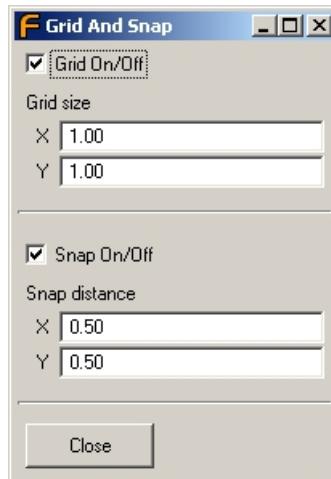
### 7.4.1 Setting Grid and Snap



You can manipulate the Grid and Snap settings in the *Control Editor* view by selecting the **Control Editor Grid/Snap** button from the *Tools* menu or the *Control Tools* tool bar. The Grid and Snap dialog box is then opened (shown at right).



**NOTE:** *Grid and Snap options are applicable only for the Control Editor view.*



### 7.4.2 Inserting blocks



To insert control system blocks, complete the following steps:

1. Click the button that represents the block you want to insert.

**TIP:** You may have to hold down the arrow found next to a block of a similar type to access the drop-down menu with other block selections.

2. Move the cursor into the *Control Editor* view. You will see the block following the mouse. Place the block where you want it and click the left mouse button to drop it.



**NOTE:** When a block is inserted in the *Control Editor* view, it is automatically added to the *Model Manager Objects* list.

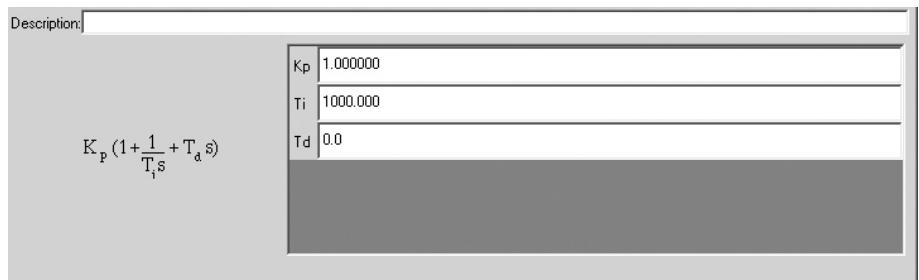
### 7.4.3 Moving blocks

You can adjust the block position by positioning the cursor over the block, pressing the left mouse button, and dragging the block to the new position. When the block is correctly positioned, release the mouse button.

Several blocks can also be moved together. To do that, press and hold the *Ctrl* key, then select the blocks you want to move and finally drag them to the new position.

#### 7.4.4 Editing block properties

Once a block is inserted in the *Control Editor* view, you can edit its properties by selecting the block and editing the properties in the Property Editor panel. For example, the equation and properties of a PID control block are shown below.

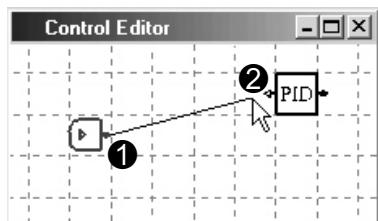


**CAUTION:** Changing a value in the Property Editor panel does not automatically apply the change. You must press the **Enter** key after editing each value to apply the change.

#### 7.4.5 Connecting blocks

Once you have placed blocks in the *Control Editor* view, you must draw connection lines between them to define the control module (shown at right). To connect two control blocks in your control module, complete the following steps:

- ① Click and drag from the arrow (output) of one control block —
- ② —to the circle (input) of another block and release the mouse button.



A connection line is drawn between the two blocks.

You can adjust the path of the line by dragging and dropping the line. Click and drag either a corner or a line segment between the corners.

##### Adding break points



This command adds lines or breaking points to connection lines. To use it, click the **Add Breakpoint** button on the *Control Tools* tool bar, then click and drag an existing line. A new corner is created.

### Removing break points



This command removes lines or breaking points on connection lines. To use it, click the **Remove Breakpoint** button on the *Control Tools* tool bar, then select one of the break points on a line. Press *Done* to confirm the selection.



**NOTE:** A line consists of at least three perpendicular line segments. It is not possible to remove line points/segments if doing so reduces the line to less than three segments.



**NOTE:** Changing the path or breakpoints of a line is for display purposes only and has no influence on control system performance.

### 7.4.6 Rotating blocks



You can rotate control blocks 180°, so that the input connections are on the right side of the block, and the output connections are on the left side of the block. To rotate a block, click the **Flip Element Direction** button on the *Control Tools* tool bar and select a block. Press *Done* to confirm the selection.



**NOTE:** This command is for display purposes only and has no influence on control system performance.

### 7.4.7 Deleting blocks or connections

To delete a block or connection line in the *Control Editor* view, complete the following steps:

1. Select the line or block in the *Control Editor* view.



**NOTE:** You can select multiple blocks at the same time by selecting them in the *Model Manager Objects* list.



2. Click the **Delete** button on the *Standard* tool bar, or the delete key on the keyboard. The selected objects are removed from the control system.



**NOTE:** When you delete a block, all connection lines to and from the block are also removed.

# Chapter 8 Mechanism Analysis

Now that you can assemble a model and implement a control system, you are ready to analyze the mechanism. This chapter describes methods of mechanism analysis in Fedem, including descriptions of setting up, starting, processing, and stopping the analyses.

Sections in this chapter address the following topics:

- Overview of Fedem analyses
- Solver tools
- Model reduction
- Model reduction in Nastran
- Dynamics analysis
- Stress recovery analysis
- Mode shape recovery analysis
- Strain rosette analysis
- Strain coat analysis
- Direct linear analysis of FE Parts
- Interaction during processing
- Deleting results
- Automated curve export from multiple result database files
- Batch execution of solver processes
- How to read error messages from the solvers
- Using simulation events

## 8.1 Overview of Fedem analyses

### 8.1.1 Model reduction

To speed up the dynamics simulation and other mechanism analyses, Fedem first performs the model reduction process. During model reduction, individual parts (FE models) in the mechanism assembly are reduced to superelements with external nodes at those points which connect parts and other mechanism elements.

Fedem uses a Component Mode Synthesis (CMS) model reduction method that replaces the internal nodal DOFs with a set of static and component modes. The model reduction process for each part generates superelement mass and stiffness matrices, which are then assembled into the system mass and stiffness matrices in the dynamics simulation. See the Fedem 7.6 Theory Guide, *Chapter 3, "Model Reduction"* for more information about the Fedem model reduction process.

You can initiate the model reduction process manually at any time (see [Section 8.3, "Model reduction"](#)). Alternatively, you may also perform the model reduction in Nastran (see [Section 8.4, "Model reduction in Nastran"](#)).

### 8.1.2 Dynamics analysis

The dynamics analysis provides the time-history solution for all displacements, velocities, accelerations, control system variables, and derived secondary quantities (such as internal reaction forces) in the mechanical system that are driven by external forces and/or prescribed displacements, velocities and accelerations.

To achieve second-order accuracy, the dynamic solution to the full system of equations is found using Newmark time-integration and Newton-Raphson equilibrium iterations at each time step. For the control system, an implicit second-order Runge-Kutta method (Lobatto IIIC) is used with Backward Euler for local error estimation. The latter is an implicit first-order method. See the Fedem 7.6 Theory Guide, *Chapter 7, "Dynamics Simulation"* and *Chapter 8, "Control System"*, respectively, for more information about the dynamics analysis of the mechanical system, and the control system analysis.

See [Section 8.5, "Dynamics analysis"](#) to set up and perform the dynamics analysis.

## Static equilibrium analysis

Before performing the dynamics simulation, Fedem may calculate an initial static equilibrium state for the mechanism model. The static equilibrium analysis establishes a starting point for the dynamics analysis, and eliminates the initial system transients such as the sudden effect of applying gravity or other external loads. In the dynamics analysis, such unbalanced forces in the initial configuration can generate undesirable effects such as vibration of the mechanism during the first time steps. See the Fedem 7.6 Theory Guide, *Section 7.8, "Quasistatic equilibrium"* for more information about the initial static equilibrium analysis.

The static equilibrium analysis determines a state for the system in which all internal and external forces are in balance in the absence of any system motions or inertial forces. All system velocities and accelerations are set to zero during this analysis.

To set up the static equilibrium analysis, see "[Initial Equilibrium tab](#)" in [Section 8.5.2](#).

## Dynamic Initial Conditions

When using the static equilibrium analysis described above, the subsequent dynamics simulation will start from a resting position with zero velocities and accelerations in all DOFs. In some cases, however, it is more relevant to start the dynamics simulation from a known velocity state in the triads and joints instead. This is possible by specifying the *Initial velocity* directly on each DOF through the respective Property Editor panels, see [Section 5.5.3, "Triad properties"](#) and [Section 5.6.2, "Joint properties"](#). This will override the global initial velocity defined in the Model Preferences dialog box, see [Section 4.9, "Model preferences"](#).



**TIP:** If the initial condition is characterized by a uniform velocity throughout the model, except for a few triads or joints where the velocity is different, it is practical to specify the "background" velocity in the Model Preferences dialog box, and use the Property Editor panel only for those Triads/Joints having a different initial velocity.



**CAUTION:** Initial velocities should be used with care such that the velocity state specified this way is consistent throughout the model. If that is not the case, fictitious transients may occur in the first time steps of the analysis.

When using initial conditions, it is recommended to specify velocities for the joint DOFs also. The initial velocities at the slave triad of the joint will then be computed automatically from the constraint equations of the joint. However, it is possible to instead specify the initial velocity in the slave triad (and leave the joint DOF initial velocities to zero), if the joint

DOF velocity is not known. Fedem will then derive the corresponding joint velocities by inverting the constraint equation of the joint.

Sometimes during the creation of a complex model, it is useful to test the dynamics when all initial conditions are off (equal to zero). To facilitate this without having to manually remove all the defined initial conditions in the model file, the solver option `-ignoreIC` may be specified instead (in the Additional Solver Options dialog box, *Dynamics Solver* field, see [Section 8.2.3, "Additional solver options"](#)). All defined initial conditions will then be ignored (assumed zero).

If you do a static equilibrium analysis prior to the dynamics simulation, any initial conditions specified will not affect the static equilibrium analysis itself. However, the subsequent dynamics simulation will then start from a configuration that is in static equilibrium, but with a given non-zero velocity state.

### Modal analysis

The dynamics analysis gives you the option of calculating the eigenmodes at different mechanism positions during the simulation. These eigenmode solutions can then be used to expand the model's mode shapes for later use in animations. See the Fedem 7.6 Theory Guide, *Section 9.6, "Eigenvalue results"* for more information about the modal analysis.

To specify the parameters for modal analysis, see "[Eigenmode tab](#)" in [Section 8.5.2](#).

### 8.1.3 Stress recovery

After performing the dynamics analysis, a stress analysis can be conducted on the mechanism. The stresses, strains and elastic displacements can then be calculated at different time steps and/or mechanism positions of the dynamics analysis. See the Fedem 7.6 Theory Guide, *Section 9.4, "Finite element stress analysis"* for more information about the stress analysis. See [Section 8.6, "Stress recovery analysis"](#) to setup and perform the stress analysis.

### 8.1.4 Mode shape recovery

A mode shape recovery analysis enables you to expand the system mode shapes calculated during the dynamics simulation. These mode shapes can later be animated. See [Section 8.7, "Mode shape recovery analysis"](#) to setup and perform the mode shape analysis.

### 8.1.5 Strain rosette recovery

Fedem enables you to recover strains and stresses from virtual strain gages which are defined in a separate file. The results are similar to those from real strain gages. See the Fedem 7.6 Theory Guide, *Section 9.5, "Virtual strain gauges"* for more information about the strain rosette analysis. See [Section 8.8, "Strain rosette analysis"](#) to setup and perform the strain rosette analysis.

### 8.1.6 Strain coat recovery

The strain coat recovery process calculates the stresses and strains on the strain coat elements in the model, in a similar manner as in the strain rosette analysis. This is primarily used as input for subsequent fatigue calculations. See [Section 8.9, "Strain coat analysis"](#) to setup and perform the strain coat analysis.

### 8.1.7 Linear analysis

Instead of doing a full dynamics simulation with subsequent recovery, it is also possible to perform a direct linear FE analysis on individual parts. This is handy if you only want to asses the static properties of a part before using it in a larger mechanism simulation. The loads and boundary conditions then need to be defined in the FE model file. See [Section 8.10, "Direct linear analysis of FE Parts"](#) to perform such linear analyses.

## 8.2 Solver tools

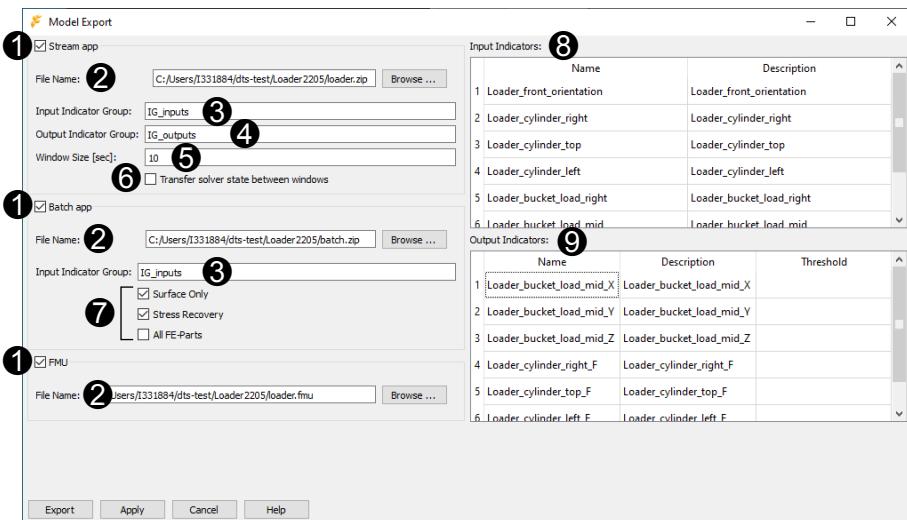
Several dialog boxes and tools can be utilized to control the calculation processes with Fedem. This section describes these general solver tools.

### 8.2.1 Model export for external simulation

In Fedem R7.6, the simulation of Fedem models outside the Fedem Desktop application itself is facilitated. This can be done by exporting the model as a *Digital Twin app*, or as a *Functional Mock-up Unit* (FMU).

A Digital Twin app is a zip-file containing all files related to the current model, in addition to some support files for executing the model in the cloud using EPD Connected Products. The FMU is also a zip-file, which in addition contains shared object libraries interfacing the dynamics solver based on the Functional Mock-up Interface (<https://fmi-standard.org>).

To open the **Model Export** dialog box (shown below), select **Export -> Export Digital Twin...** from the **File** menu.



- ① Activate any of these three toggles to enable the export of the current model as a *Stream app*, a *Batch app*, and a *FMU*, respectively. The fields in the respective frames will be active only if the associated toggle is enabled.
- ② *File Name:* - For each of the three app types, you can click the *Browse...* button to select the name of the zip-archive that will contain the exported model.
- ③ *Input Indicator Group:* - Specifies the group name of input indicators that will be used in the exported *Stream-* or *Batch app*. The names specified here need to be reflected in the cloud simulation setup in *EPD Connected Products*.
- ④ *Output Indicator Group:* - Specifies the group name of output indicators that will be used in the exported *Stream app*. The name specified here needs to be reflected in the cloud simulation setup in *EPD Connected Products*.
- ⑤ *Window Size [sec]:* - Specifies the length of each time window (in seconds) for the *Stream app*.
- ⑥ *Transfer solver state between windows* - Activate this toggle if each time window should be restarted using the final state of the previous window as starting point. If not activated, each time window will start from the modeling configuration.

**7** Toggles for exporting *Batch apps*:

- *Surface Only* - Activate this toggle to represent all FE parts in the model by surface elements only in the visualization model. If not activated, the visualization model may be larger since it will also contain internal element interfaces (invisible) between the solid finite elements.
- *Stress Recovery* - Activate this toggle to let the app perform deformation- and von Mises stress recovery for the FE parts that have the *Perform stress recovery during dynamics simulation* toggle enabled in the "*Advanced tab*" of the Part property editor panel.
- *All FE-Parts* - Activate this toggle to let the app generate a visualization for all FE parts in the model. If not activated, only the parts having the *Perform stress recovery during dynamics simulation* toggle enabled in the "*Advanced tab*" of the Part property editor panel will be included in the visualization. When the *All FE-Parts* toggle is enabled, FE parts which are not toggled for recovery will be represented by their rigid body motion only.

**8** *Input Indicators*: - This table lists all "*External function*"s in the model (their *Tag* and *Description*, respectively. They will define the content of the named *Input Indicator Group*.

**9** *Output Indicators*: - This table lists all "*Functions*" in the model, that have the *Use as output sensor* toggle enabled in the property editor panel. Their *Tag* and *Description*, respectively, are listed, in addition to the description of the *Threshold* settings, if enabled.

Pressing the *Export* button will perform the model export for all of the three app types which have the respective activation toggle enabled.

### 8.2.2 Solvers tool bar

The *Solvers tool bar* (shown below) contains the commands to set up and start each of the mechanism analysis processes, including the post-processing of individual mechanism parts. The tool bar is organized from left to right in the order of logical task performance, i.e., the dynamics simulation (including model reduction, if needed) is performed first, then the stress recovery, mode shape recovery, and so on.



The setup commands (**Dynamics Solver...**, **Stress Recovery Setup...**, etc., and **Additional Solver Options...**) enable management of all analysis options. Each of the *Solver* commands are described in the following sections of this chapter.



**TIP:** To access all commands on the Solvers tool bar, click and hold down those buttons with an arrow (▼) next to the icon.



**NOTE:** You can also access all Solver tools from the Solve menu.



Once you have set up the parameters for each of the solvers you would like to execute (as described later in this chapter), click the **Solve All** button on the Solvers tool bar (or *Solve* menu) to execute all analyses automatically in consecutive order.

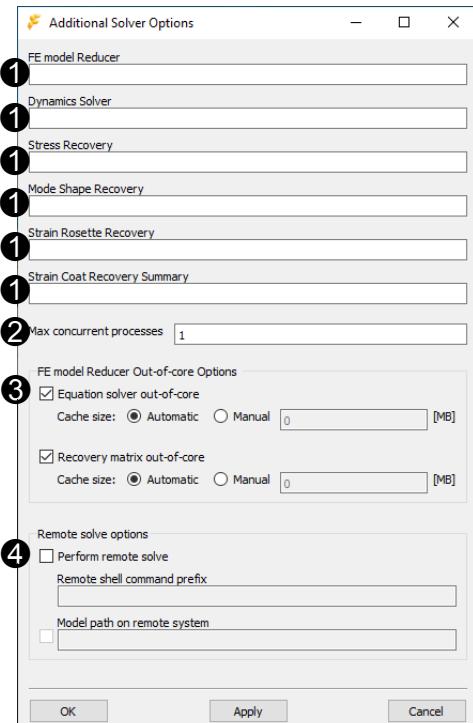
### 8.2.3 Additional solver options

In the Additional Solver Options dialog box, advanced users can fine tune the behavior of the solver modules through options that are not available elsewhere from the respective setup dialog boxes or the various Property Editor panels.



Select **Additional Solver Options...** from the *Solve* menu to open this dialog box (shown below).

- ① In each of the first six fields, you may enter text strings of command-line options for the respective solver module. Refer to [Appendix C, "Command line options"](#) for a complete list of options for all Fedem modules.
- ② You can set the maximum number of concurrent processes that will be run during a simulation task. This is useful if you have a multi-processor machine and want to run several part reductions or recovery processes in parallel.
- ③ You can fine-tune the memory usage of the FE model Reducer through these options. See ["Optimizing the FE model Reducer memory usage"](#) below for further details.



- ④** You can specify a command prefix to be applied on all solve tasks. This can be used to launch the simulation tasks on another computer in your local network, than the Fedem UI is executed on. See "["Running solver processes on a remote computer"](#)" below for further details.



**CAUTION:** Many of the solver options listed in [Appendix C, "Command line options"](#) may already have been provided to the corresponding solver through their respective setup dialog boxes. Specifying any such option also in the Additional Solver Options dialog box will then override the setting in the solver-specific dialog box and should therefore be avoided. Consult Fedem technical support if you are in doubt on the usage of a particular command-line option.



**NOTE:** If you mis-spell a command-line option in the Additional Solver Options dialog, or specify options that do not exist, the solver process will run as if the invalid options were not specified. A warning for each unrecognized option is issued in the [Output List](#) in that case, after the solver process has terminated.

### Optimizing the FE model Reducer memory usage

Perhaps the most memory critical solve process in Fedem is the FE model reduction for large models. On 32-bit platforms, the amount of in-core memory that one process may address is 2 GB (usually the practical limit is lower due to other processes sharing the same CPU). In Fedem, a linear equation solver is used in the FE model Reducer which works out-of-core when necessary. This makes it possible to solve much larger FE models on a 32-bit platform than would be possible using an in-core solver. The equation solver reserves an in-core buffer (cache) for the numerical data of a certain size, and goes out-of core only when this buffer is not large enough. The performance of the equation solver depends on the size of this buffer and it may therefore be optimized by fine-tuning this size.

The default is to let the Reducer automatically set the size of the in-core buffer (as shown to the right). It then reserves a fixed percentage of the free memory currently available (the actual size is written to the `.res` file). This size may be overridden by switching to **Manual** and entering the desired cache size in the corresponding field.

FE model Reducer Out-of-core Options	
<input checked="" type="checkbox"/> Equation solver out-of-core	
Cache size:	<input checked="" type="radio"/> Automatic <input type="radio"/> Manual <input type="text" value="0"/> [MB]
<input checked="" type="checkbox"/> Recovery matrix out-of-core	
Cache size:	<input checked="" type="radio"/> Automatic <input type="radio"/> Manual <input type="text" value="0"/> [MB]

You may also switch off the out-of-core feature completely by toggling off the **Equation solver out-of-core** option. The Reducer will then abort if the problem does not fit in core.

There is also a similar set of memory usage options that affects the displacement recovery matrix, and they work in a similar way. Using the **Manual** setting here corresponds to the *-Bramsize* command-line option.



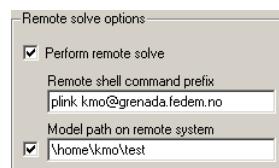
**NOTE:** Since Fedem R7.2, only the *in-core* equation solver has been available in the FE model Reducer. The *out-of-core* equation solver settings discussed above will therefore be ignored, whereas the options for the recovery matrix still apply. The documentation of the *out-of-core* equation solver settings is still retained here for completeness. The *out-of-core* feature is anyway less relevant for the current 64-bit platforms, where *in-core* memory is less limited.

### Running solver processes on a remote computer

If you are using a workstation connected to a file server in a local network together with other computers, it may be advantageous to perform the simulation tasks on one of the other computers, such that the local workstation can allocate its resources fully to the Fedem UI process.

To enable such remote solving, use the **Perform remote solve** toggle in the *Remote solve options* part of the Additional Solver Options dialog box (shown to the right). Enter the appropriate

*Remote shell command prefix* needed to run the solver command on the remote computer, and optionally the *Model path on remote system*. The latter is necessary when, e.g., your local workstation is a Windows PC and the remote computer is a UNIX machine. You will then need to specify the path to the current model file, as it looks from the UNIX computer.



**CAUTION:** When specifying a remote shell command prefix, the input files needed by the solver tasks are still created locally within the Fedem UI and not explicitly copied on to the remote computer. Thus, a remote execution will work only if the local and remote computers use a shared file system. Similarly, it is assumed that the Fedem UI on the local computer can access the output files created by the solver task directly from where they are written by the remote computer.



A server program accepting remote shell commands must be running on the specified remote computer for this feature to work (e.g., sshd or rshd). If using ssh (or the equivalent windows client program plink), you will also need to define proper identification keys in your login directory such that you are able to login to the remote computer without being prompted for a password. Please ask your system operator for assistance on such issues. For further information on the ssh and PuTTY/plink programs, consult, e.g., <http://www.openssh.com> and <http://www.chiark.greenend.org.uk/~sgtatham/putty>.

### 8.2.4 Controlling placement of temporary files

Some of the Fedem solvers use temporary files during computations, which are automatically deleted upon completion of the process.

On UNIX systems, these files are placed in the directory pointed to by the environment variable TMPDIR, if set. If TMPDIR is not set or points to a non-existing directory, they are placed in the directory /var/tmp instead. On Windows, they are placed in the directory pointed to by the environment variable TMP, if set. If TMP is not set or points to a non-existing directory, they are placed in the directory C:\ instead.



**NOTE:** Some of the temporary files may become very large. Make sure that the TMP or TMPDIR variable points to a directory with sufficient amount of free disk space.

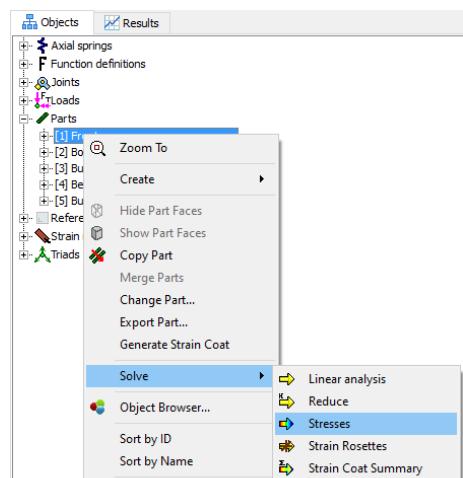
### 8.2.5 Part- and group-wise solving

Fedem has the capability of running some of its solvers on individual parts of the mechanism (FE Parts and element groups). This is beneficial when dealing with big models where the solver execution may be both time- and disc space consuming.

Note that these solver tasks act upon each part independently, and do not affect the global response in any way. The user is encouraged to identify critical parts of the model and recover just the results he wants on these parts. Be aware that each time you recover results they are added to the result database, regardless of their previous existence.

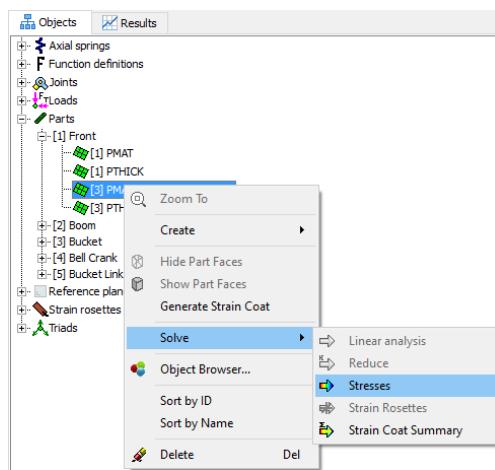
#### Part-wise solving

Solver processes that may be run on parts individually are: Linear Analysis, Reducer, Stress recovery, Strain Rosette recovery and Strain Coat recovery. A part-wise solve process is started by right-clicking on a part in the *Objects* list of the Model Manager panel, choosing **Solve**, and then the wanted process (see illustration to the right). Note that you may multi-select parts in the *Objects* list to solve for two or more parts simultaneously. As always, when trying to run the Reducer, a part will only be reduced if necessary.



### Group-wise solving

Solver processes that may be run on individual element groups are: Stress recovery and Strain Coat recovery. A group-wise solve process is started in a manner similar to that of a part-wise process (see Illustration to the right). Right-click the group for which you want to solve and then choose the wanted process. Multi-selection is possible to solve for two or more groups simultaneously. For further information on element groups, refer to [Section 5.2, "Element groups"](#).



**NOTE:** When running Strain Coat recovery on individual parts or element groups, a set of strain coat elements are automatically created on all shell and solid elements in the current selection, unless such elements have been created in a previous run (see also [Section 8.9.4, "Strain coat recovery on element groups or individual parts"](#)).



**NOTE:** When solving on individual parts or element groups, the current settings in the corresponding solver setup dialog box are used.

### 8.2.6 Viewing the progress of long-duration analyses

When solving a large model that takes a considerable amount of time, it is often informative to know exactly how far in the simulation process we have reached at any time. This can be done simply by viewing a res-file that is continuously being written by the running solver in the Result File Browser (see [Section 10.2, "Result File Browser"](#)). When such a res-file is selected, the Info view is automatically scrolled to the bottom of the file and then continuously updated as the file is being written by the solver.



**TIP:** For the model reduction and all of the recovery modules, a dedicated progress file called `progress_info.res` is created. This file is updated more frequently than the conventional res-file associated with the process. Viewing this file while the solver is running will therefore give you the best update on its progress.

It is often wise to also keep an eye on the *Output List* view while a solver process is running, as important messages produced by the solver (errors, warnings and notes) are written here while the process is running (see also [Section 8.14.2, "How to read error messages from the solvers"](#)). All such

messages are also written to the res-file, but in the *Output List* they are prefixed by the solver name and the process ID in brackets, e.g.

```
fedem_solver [2716]: Started.
fedem_solver [2716]: Warning : Indication of a poor centripetal force on the rotating Link 3
fedem_solver [2716]: Try distributing the triads better or use centripetal moment correction.
fedem_solver [2716]: Finished. Wall time elapsed: 00:04:07
```

The first message of a solver process is always "Started." and the last message is "Finished." followed by the consumed wall time. The process ID number is used to distinguish messages from possibly multiple simultaneously running reduction or recovery processes.

## 8.3 Model reduction

The FE model reduction process requires no user setup apart from the settings found in the *Reduction Options tab* of the Property Editor panel for each part (see [Section 5.1.5, "Part properties"](#)). Some of these options are discussed in further detail in the sub-sections below.

### 8.3.1 Starting the model reduction

Model reduction is performed automatically for the parts needing it when you start the dynamics simulation. Fedem determines automatically which parts need to be reduced based on the triad configuration and the connection to the rest of the model. It also checks whether any of the settings in the *Reduction Options tab* that affects the results have been changed since the last reduction of that part.



You can also initiate the model reduction process manually at any time. To do this, simply select **Reduce All FE Parts** from the *Solve* menu. You may also initiate the model reduction for only one, or a selection of parts, by using the *Part-wise solving* command (see [Section 8.2.5, "Part- and group-wise solving"](#)).



**NOTE:** If element calculations fail during model reduction due to bad element shapes, etc., all bad elements in the part are reported before the model reduction process exits.



**NOTE:** Once a part is reduced, it will not be reduced again, even if the Reduce All Parts button is clicked again, unless the part has been modified in the meantime (by for instance, adding or removing external nodes or altering the material properties).

### 8.3.2 Using component modes

Fedem uses a Component Mode Synthesis (CMS) model reduction method that replaces the internal nodal DOFs with a set of static and component modes. The static modes corresponds to the external nodal DOFs (i.e., at the Triads attached to the part), whereas the component modes are calculated as the eigenmode shapes of the part with all external nodes fully constrained.

Component modes describe the internal vibrations in the part. You should normally include a sufficient number of modes such that the frequencies within the time step size used in the dynamics simulation are covered. The frequencies of the computed component modes are found in the output file `fedem_reducer.res` which can be viewed using the *Result File Browser* (see [Section 10.2, "Result File Browser"](#)).

The number of component modes is specified in the Property Editor panel for the selected part (see "[Reduction Options tab](#)" in [Section 5.1.5](#)). The default is 0 (no component modes), i.e., only static modes are used. If the internal vibrations are not important for the overall response, you can save computation time by using static modes only. More details on component modes and CMS model reduction can be found in the Fedem 7.6 Theory Guide, [Section 3.2, "Component mode synthesis reduction"](#).

### 8.3.3 Using lumped mass matrix

To increase the computational efficiency of the model reduction process, a lumped mass matrix approach is used by default. The FE mass matrices are then represented by one diagonal matrix each, where each diagonal term is the sum of the associated row (or column) of the corresponding consistent mass matrix. The assembled mass matrix of the part will then also be diagonal, except for some off-diagonal terms created by the constraint equations<sup>1</sup>, if any. Thus, using lumped mass will reduce the memory requirements significantly compared to using a consistent mass matrix, and will also speed up the reduction process as some of the linear algebra operations involved are simplified.

- 
1. You have constraint equations in the part if it contains RBAR, RGD, WAVGM (RBAR, RBE2, RBE3 in Nastran terminology) and/or BEAM elements with end release. Thus, as the number of such elements in the part increases, the advantage of using lumped mass will decrease.



**NOTE:** To some degree, using lumped mass does affect the results, as it represents a simplification of the FE model. The difference in the results may be significant for parts with a coarse FE mesh. However, as the FE mesh is refined, the difference becomes smaller. Therefore, it is recommended to use consistent mass on parts not having a sufficiently fine FE mesh. If you are in doubt, try a test run using both consistent and lumped mass and compare the results.

The use of consistent mass matrix is toggled on/off for the selected part through a button in the Property Editor panel (see "[Reduction Options tab](#)" in [Section 5.1.5](#)). The default is off for new parts imported into a model.

### 8.3.4 Handling singularities during the model reduction

In complex FE modeling, it is inevitable not to have one or more defects in the FE model from time to time. In the model reduction process, this will typically result in a singular mass and/or stiffness matrix. Fedem recognizes two types of singularities, which are handled slightly different in the matrix triangularization (factorization):

- DOFs that have not received any stiffness/mass contribution at all and thus have an exact zero pivot before the triangularization is started, are detected *a priori*. The zero pivot is then replaced by the value 1.0, which implies that the singular DOF is constrained to zero.
- DOFs that initially have a non-zero pivot, but are reduced to a value close to zero during the triangularization, will receive some small value on the diagonal, allowing the triangularization to continue for the detection of other potential singularities. The judgment on whether such singularities have occurred is based on the user-provided "[Singularity tolerance](#)" (see below).

When the reduction process has terminated, a list of all singularities is written to the *Output List* and the *.res* file. Each singular DOF is here identified by the node ID and local DOF number. If only singularities of the first kind are detected, the reduction process completes successfully and it should be safe to use this part in a dynamics simulation. However, if one or more singularities of the second kind are detected, the reduction is aborted with no results, and the FE model has to be manually fixed before a new attempt is made.



**CAUTION:** If more than one singularity of the second kind are detected, there is a possibility that some of them, except the first one, are fictitious. On the other hand, there is also a possibility that not all singularities are detected. Thus, after having fixed all the reported singularities manually, other singularities may be revealed in the next run. This behavior is due to the insertion of a small value on the diagonal which actually changes the mechanical property of the FE model.

It is possible to switch off the above treatment of singularities during model reduction. This is done by specifying `-singularityHandler 0` as an additional option to the FE model Reducer (see [Section 8.2.3, "Additional solver options"](#)). Only the first occurring singularity, regardless of its kind, will then be written to the *Output List* and `.res` file, and then the reduction process will abort.

### Singularity tolerance

The criterion used to determine if a stiffness or mass matrix is singular during the reduction is specified through a threshold value in the Property Editor panel for the part (see ["Reduction Options tab"](#) in [Section 5.1.5](#)). The default value is 1.0e-12. If the ratio between the current and previous value of a diagonal matrix element becomes less than the threshold during the factorization, the matrix is assumed to be singular.

The default threshold value is usually sufficient for well-conditioned FE models. However, there might be situations where you have a natural high ratio between the stiffness (or mass) properties in different parts of a model. In this case you may need to reduce the value to avoid a false error exit. If you do get a singularity exit but is quite sure your model is sane (although not that well-conditioned), you should check the actual diagonal decay value which is written in the `.res` file. You may then try changing the *Singularity criterion* to a value less than this value and rerun the model reduction. Note that the lowest admissible value is 1.0e-20. Any value lower than that will be ignored and 1.0e-20 will be used.

### Negative pivots

A model reduction may also reveal negative pivot elements in the stiffness or mass matrix. This may happen if, e.g., the part contains several poorly shaped elements (especially shell elements). However, since the linear equation solver is able to handle negative definite matrices, the reduction is not aborted when only negative pivots are encountered. Nevertheless, it is good practice to go over the FE model once again and check for bad elements if you get warnings on negative pivots.



**WARNING!** *Using a reduced part with many negative pivots in its stiffness matrix may lead to instabilities in the subsequent Dynamics simulation, and should be avoided. Only a few negative pivots in a large part is normally not a problem, however. That has at most only local influence on the results in the vicinity of the elements with the negative pivots.*

### 8.3.5 Eigenvalue analysis of the reduced parts

To assess the dynamic properties of the reduced FE part matrices, you can perform an eigenvalue analysis of the reduced system, i.e.

$$(\mathbf{K} - \omega^2 \mathbf{M})\phi = 0$$

where **K** and **M** denote the reduced stiffness and mass matrices. Since the reduced system does not include any constrained DOFs<sup>1</sup>, the first six eigenvalues should always be zero. The remaining eigenvalues can then be used to assess the dynamic properties of the reduced part.

The  $n$  lowest eigenfrequencies, i.e., the quantities

$$\frac{\omega_i}{2\pi}, \quad i = 1 \dots n$$

for each reduced part can be written to the corresponding `.res` file by specifying `-nevred n` in the *FE model Reducer* field in the Additional Solver Options dialog (see [Section 8.2.3, "Additional solver options"](#)). The eigenfrequencies are presented in Hertz. The default value of this option is 12 for all parts. However, the option is not effective for massless parts. For other parts, the option may be turned off by specifying `-nevred 0`.

### 8.3.6 Visualization of eigenmode shapes from the model reduction

To further verify the results of the reduction process and to increase the understanding of the part's dynamics properties, it is often useful to visualize the computed mode shapes of the part. This is possible if you toggle on *Expand mode shapes* in the *Reduction Options tab* of the Property Editor panel for the parts in question, before they are reduced (see [Section 5.1.5, "Part properties"](#)). Both the component mode shapes (see [Section 8.3.2, "Using component modes"](#)) and the mode shapes associated with the eigenvalues of the reduced system (see [Section 8.3.5, "Eigenvalue analysis of the reduced parts"](#)) will then be computed during the model reduction, and be subsequently available for viewing in an Eigenmode animation (see "*Eigen Modes tab*" in [Section 9.5.2](#)).



**TIP:** It is particularly useful to study the mode shapes of the reduced system, if you for instance get more than six modes with (close to) zero eigenfrequency. That is usually due do an internal mechanism in the part caused by an error in the FE-mesh and can be revealed when animating the corresponding mode shape.

1. DOFs that should be constrained are retained during the part reduction and are constrained only during the system analysis by the dynamics solver.



**TIP:** If you get less than the required six zero rigid mode modes in the reduced system for a part, that can be caused by some over constraining in the FE model (due to bad modeling, etc.). By animating the six first mode shapes you can then see which rigid body modes are present and which are not, and also see the mode shapes that replaced the missing rigid body modes. All this might then give a hint towards the actual model error or weakness.

### 8.3.7 Reduction of applied load vectors

Along with the reduced mass- and stiffness matrices, the FE model Reducer always computes reduced (unit) gravity load vectors of the part, which are applied as a constant static load in the direction of the defined gravitation vector in the dynamics simulation.

In addition, a set of load vectors corresponding to defined load cases in the FE data file are computed, when such load definitions are present. This includes both concentrated point loads and distributed surface loads on shell and solid elements.

These reduced load vectors may then be assigned time history scaling functions in the Property Editor panel before the dynamics simulation is run (see "[Reduced Loads tab](#)" in [Section 5.1.5](#)).

## 8.4 Model reduction in Nastran

As an alternative to the Fedem Model Reduction, the model reduction may also be performed in Nastran. Nastran supports a similar CMS reduction procedure as in Fedem, and is able to export the reduced mass- and stiffness matrix as binary files that may be imported into Fedem. The advantage of doing the model reduction in Nastran is that you then have a broader range of element and material properties available for use on the part level, such as orthotropic materials, composites, etc.

### 8.4.1 Nastran DMAP

The Nastran model reduction is performed using a *DMAP* script to facilitate the generation of the reduced matrices needed in Fedem. A Nastran DMAP is a script program (kind of API and programming language) that modifies the execution of the Nastran solver. The script is model independent and must be *included* in the Nastran bulk data file when Nastran reduction is desired. The script, called *nastran\_dmap.dat*, is located in the Template folder of the Fedem installation. Refer to the Nastran documentation for further details on the DMAP script language.

### 8.4.2 Nastran bulk data entries for CMS reduction

The Nastran bulk data file must contain some commands, in addition to the entries describing the FE model itself, when it is going to be reduced in Nastran. Before the CEND keyword, the following commands must be added to define the output files, solution type and the DMAP script:

- Define OP2-files used to store the reduced mass (m), stiffness (s) and gravity (g) matrices from Nastran (<name> is here some arbitrary string identifying this part):

```
ASSIGN, output2='<name>_m.op2',UNIT=71
ASSIGN, output2='<name>_s.op2',UNIT=72
ASSIGN, output2='<name>_g.op2',UNIT=73
```



**Note:** The OP2-files are automatically converted to .fmx files when the Nastran bulk data file is imported into Fedem, and the part is recognized as Reduced.

- Specify modal analysis:

```
SOL 103
```

- Include DMAP script:

```
INCLUDE 'nastran_dmap.dat'
```

After the BEGIN BULK keyword, the following commands must be added to perform the Nastran CMS-reduction:

- If you need to attach triads or joints to shell element nodes, you need to turn on the drilling dof on the Nastran shell elements:

```
PARAM, K6ROT, <value>
```

<value> = 10 or a small value is recommended. (If it is set to 0 the drilling dof is omitted, which is not what is desired). The value is used to calculate an artificial stiffness that might influence the overall stiffness. This might introduce undesirable effects, and the resulting component modes should therefore be compared with component modes calculated without this artificial stiffness.

- To get consistent instead of lumped mass representation the following bulk data command may be added:

```
PARAM, COUPMASS,1
```

If coupled mass is not used, the DMAP script can be simplified since  $M_{ie} = M_{ei} = 0$ .

- Definition of static modes (external dofs). Use ASET or ASET1 entries to define the external dofs. To get 6-DOF Triads, all nodes in the ASET (active dof set) must have 6 active DOFs. A typical bulk entry is:

```
ASET, node_id1, 123456, node_id2, 123456, ...
```

### Adding component modes (generalized dofs)

- If component modes (generalized dofs) are desired, the following bulk entries must be specified:

```
EIGRL,1,,,<idn>
SPOINT,<id1>,<id2>...<idn>
QSET1,0,<id1>,<id2>...<idn>
```

Here <idn> is the number of component modes.

#### 8.4.3 Recovery of parts reduced in Nastran

When the model reduction is performed in Nastran, you also have to perform the part-level recovery in Nastran. Stress Recovery, Mode Shape Recovery, etc., can not be performed by Fedem for such parts.

Some dynamics solver results must be exported to bulk data format files when part-level recovery is to be performed in Nastran. To do this, specify “-savelnc3 <inc>” as *Additional solver options* for the dynamics solver, where <inc> is the time increment between each time step you want to perform recovery for. You will then get one file for each Nastran-reduced part for each time step in the results database. These files have names on the form “*adisp\_<part>\_t<istep>.bdf*”, where <part> is the base name of the FE model file and <istep> is the time step number, and can be loaded into Nastran together with the FE model file for doing the recovery there.

## 8.5 Dynamics analysis



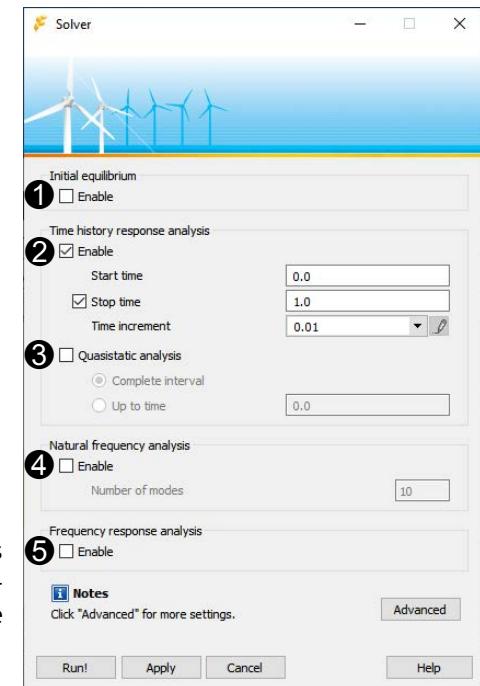
To control the dynamics analysis parameters, click the **Dynamics Solver** button on the *Solvers* tool bar (or select **Dynamics Solver (Basic Mode)...** from the *Solve* menu). The Dynamics Solver dialog box (shown below) will then appear and allow you to adjust the simulation setup.

In-depth information about the time integration algorithm employed by the dynamics solver may be found in the Fedem 7.6 Theory Guide, *Chapter 7, "Dynamics Simulation"*.

### 8.5.1 Dynamics Solver (Basic Mode)

In its basic mode, the Dynamics Solver dialog box contains the following:

- ① *Initial equilibrium* - Enables a static equilibrium analysis to move the mechanism to a resting position before simulating the dynamics, see "["Static equilibrium analysis"](#)" in [Section 8.1.2](#).
- ② *Time history response analysis* - Enables the dynamics simulation of the time history response, with specification of the *Start*, *Stop* and *increment* size of time domain.
- ③ *Quasistatic analysis* - Enables a quasi-static simulation, either for the complete time interval or up to a specified time. In this part of the analysis, all velocity- and acceleration terms (i.e., the damping and inertia forces with associated tangent contributions) are set to zero.



The *Quasistatic analysis* can be used in place of (or in addition to) the *Initial equilibrium* analysis to obtain a proper start configuration for the subsequent dynamics simulation, in case the modeling configuration is so far off equilibrium that more than one load step is needed to obtain a converged configuration.

- ④ *Natural frequency analysis* - Enables an eigenvalue analysis to be performed to compute a specified number of modes, see "[Modal analysis](#)" in [Section 8.1.2](#). The eigenvalue calculation is performed on the initial configuration, and then at time intervals as specified in the *Eigenmode tab* of the *Advanced* solver dialog box (see [Section 8.5.2, "Dynamics Solver \(Advanced Mode\)"](#) below).
- ⑤ *Frequency response analysis* - Enables a frequency response analysis to be performed during the time history response simulation, if the model contains Triad- or Joint DOF loads or prescribed motions which have been defined as frequency domain inputs, see [Section 5.5.3, "Triad properties"](#) and ["Joint variable properties"](#) in [Section 5.6.2](#). The toggle is inactive if no such loads or motions exist in the model. See [Section 8.5.3, "Frequency response analysis"](#) for more details on performing frequency response analyses.

You can click the **Run!** button to start the dynamics solver, or the **Advanced** button, if more specialized setup is needed first (see below).

### 8.5.2 Dynamics Solver (Advanced Mode)

The advanced settings for the Dynamics Solver are placed on six tabs in the dialog box labeled *Time*, *Integration*, *Tolerances*, *Eigenmode*, *Initial Equilibrium*, and *Output*, respectively, and are described in the following.

All six tabs have in common that they have a **Basic** button that will take you back to the basic mode of the Solver dialog box (see [Section 8.5.1, "Dynamics Solver \(Basic Mode\)"](#)). If you then click the corresponding **Advanced** button in the basic Solver dialog box, you will return to the tab of the advanced dialog box that you just left.

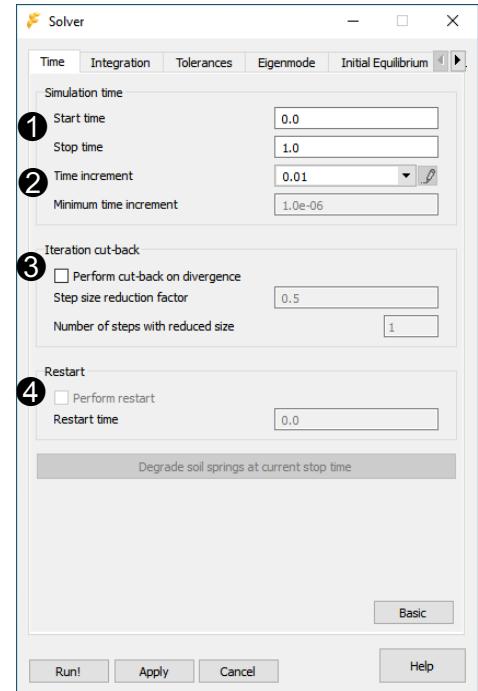
## Time tab

The time domain of the dynamics simulation is controlled through the following parameters:

- ❶ You can define the *Start time* and the *Stop time* of the dynamics simulation.
- ❷ You can define the size of the *Time increment* to be used by the time integration algorithm. In addition to a constant value, you may also select a *Function* or a *Time history input file* from the pull-down menu, in order to obtain a varying time increment size (see [Section 5.11, "Functions"](#)). In that case, the *Minimum time increment* is used as a lower bound on the step size.
- ❸ You can enable/disable the use of *Iteration cut-back* when the dynamics simulation diverges, and adjust the *Step size reduction factor* defining the size of the new time step to use in the cut-back, and the *Number of time steps with reduced size* before the normal step size is resumed. If the cut-back iterations also diverge, another cut-back is attempted by applying the *Step size reduction factor* again. This procedure is then repeated until convergence is obtained or the *Minimum time increment* is reached. In the latter case, the simulation is aborted.
- ❹ You can enable a *Restart* simulation if you already have some simulation results, and specify the *Restart time* defining the time step to restart from. If the specified time does not match an existing time step, the closest step after the specified time is used.



**NOTE:** In a restart simulation, you are allowed to adjust any of the other Solver Setup parameters and options, as well as adjusting the additional solver options for the Dynamics solver (see [Section 8.2.3, "Additional solver options"](#)). However, you can not change any properties of the mechanism model itself.



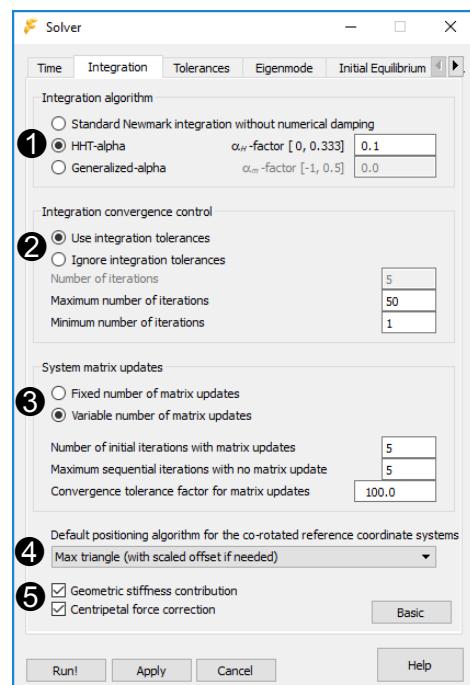
Restarting a Dynamics simulation is very handy if you discover you need to continue a simulation that was terminated abnormally, or you just want a longer event than was originally defined. You may restart a simulation as many times you wish.

Each restart adds a new set of result files to the existing results database such that subsequent post-processing and recovery runs can be conducted over the overall time domain spanned by both the initial run and the restart(s). If the time domain of the individual runs overlap, only the latest produced results will be used in post-processing and recovery. See [Section 10.2.3, "Result files from restart simulations"](#) for more information on the management of results from restart simulations.

### Integration tab

You can optimize numerical performance of the time integration by adjusting the following parameters:

- ① You can select between three different time integration algorithms (*HHT-alpha* with  $\alpha_H = 0.1$  is recommended). Refer to the Fedem 7.6 Theory Guide, Sections 7.2, 7.3 and especially 7.4 “*Newmark integration with numerical damping*” for details on the different time integration schemes available.
- ② You can enable/disable the use of *Integration tolerances*, and specify the *Maximum* and *Minimum number of iterations* for each time increment. When *Ignore integration tolerances* is set, the fixed *Number of iterations* is specified instead.
- ③ The nonlinear equations are solved using Modified Newton-Raphson iterations meaning that the system matrices are not necessarily recalculated in each iteration. For efficiency reasons, the number of matrix updates per time step should be as low as possible. However, if the increments in the input variables are large during a



time step, the system matrices need to be updated more often, to ensure the nonlinear iterations converge.

You may choose between *Fixed number of matrix updates* or *Variable number of matrix updates*. In either case you can also specify the *Number of initial iterations with matrix updates* and the *Maximum sequential iterations with no matrix updates*. The first number defines how many iterations in the beginning of each time step should be performed with updated matrices. If the iterations have not converged before reaching that number, the subsequent iterations are performed with a matrix update frequency defined by the *Maximum sequential iterations with no matrix updates*. However, if *Variable number of matrix updates* is chosen, a convergence based threshold is used in addition to determine when to do further system matrix updates. The factor entered in *Convergence tolerance factor for matrix updates* is then multiplied with the active convergence tolerances specified on the *Tolerances tab*. The resulting tolerance is compared to the error norms corresponding to the active convergence tolerances and the matrices are updated as long as the error norm is higher than this tolerance.

- ④ You can specify the default algorithm for calculation of the co-rotated part coordinate systems during the simulation. The selections available correspond to those of the similar pull-down menu in the *Advanced tab* of the Property Editor panel for Parts (see [Section 5.1, "Part properties"](#)). The setting here applies to all parts in the model, where the corresponding setting on the part level is "Model default".
- ⑤ You can enable/disable both the *Geometric Stiffness Contribution* and the *Centripetal Moment Correction* during the non-linear iterations.

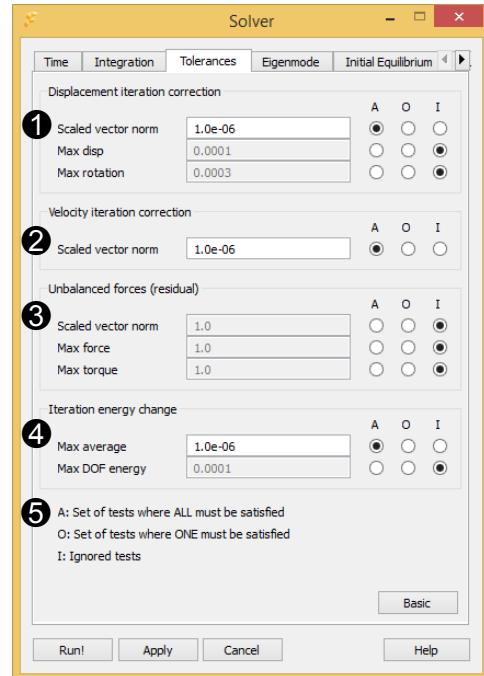
The geometric stiffness option accounts for stress stiffening in parts, and axial springs and dampers. It may thus improve the convergence of the nonlinear iterations if the model contains parts with large membrane forces or axial springs and dampers with large forces. This is because the forces alters the bending- or rotational stiffness of these elements. A tensile membrane/axial force will effectively increase the bending/rotational stiffness and a compressive force will reduce the bending/rotational stiffness. For information on the computation of the geometric stiffness on parts, see the Fedem 7.6 Theory Guide, [Section 4.4 "Superelement tangent stiffness"](#).

The centripetal moment correction option enables an improved representation of the inertia forces on parts with only a few Triads that experience high-speed rotations (see the Fedem 7.6 Theory Guide, [Section 3.3, "Inertia forces and high-speed rotation"](#) for details).

### Tolerances tab

Convergence criteria for the dynamics analysis are defined by enabling one, or more, convergence tolerances:

- ① This allows you to enable/disable and set convergence tolerances for the *Displacement iteration corrections*.
- ② You can define a convergence criterion on *Velocity iteration correction*
- ③ You can define various tolerances for *Unbalanced forces*.
- ④ You can define convergence tolerances for iteration energy changes.
- ⑤ All available convergence tests can be ignored or defined into one of two sets of tests: In set A, all tests must be satisfied for convergence to be satisfied. In set O, only one of the tests must be satisfied (in addition to all the tests in set A) for convergence to be satisfied.



The various norms used in the above convergence criteria have different dimension properties. Some are dimensionless, whereas others depend on the model units. The default values defined are suitable for the SI unit set. If you model in a different unit set (see [Section 4.9, "Model preferences"](#)), you will need to adjust some of your active tolerance values accordingly.

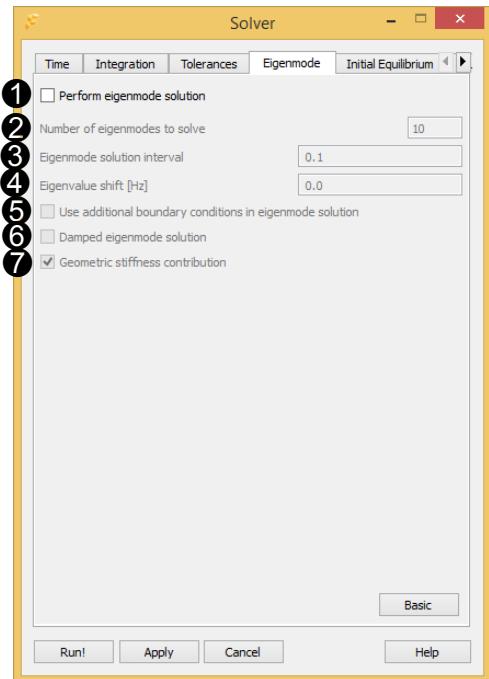
The scaled vector norm of the displacement correction is dimensionless, whereas the same norm for the velocity correction has dimension  $1/\text{[time]}$  and for the force residual it is  $[\text{force}][\text{length}]$ . The Max norms have the dimension corresponding to the quantity that they measure.

For more information about convergence criteria, see the Fedem 7.6 Theory Guide, [Section 7.3.1, "Convergence criteria"](#).

### Eigenmode tab

You can set up the calculation of eigenmode solutions (see "[Modal analysis](#)" in [Section 8.1.2](#)) by adjusting the following parameters:

- 1 This option allows you to enable/disable calculation of the eigenmode solutions.
- 2 You can specify the number of eigenmodes to be computed.
- 3 You can specify the time interval between each eigenvalue analysis.
- 4 You can specify an *Eigenmode Shift* (see the Fedem 7.6 Theory Guide, [Section 9.6.3, "Using shift when solving the eigenvalue problem"](#)).
- 5 You can enable/disable the application of the *Additional Boundary Conditions* specified for triads (see [Section 5.5.3, "Triad properties"](#)).
- 6 You can enable/disable computation of damped eigenmodes by accounting for structural damping (see the Fedem 7.6 Theory Guide, [Section 9.6.2, "Damped eigenvalue problem"](#)).
- 7 You can enable/disable the *Geometric Stiffness Contribution* in the eigenvalue analyses (see the similar option in the [Integration tab](#)). If the mechanism contains structural members that experience large tensile or compression forces at certain time steps, the geometric stiffness contribution may have a significant effect on the accuracy on the computed eigenvalues at those time steps.

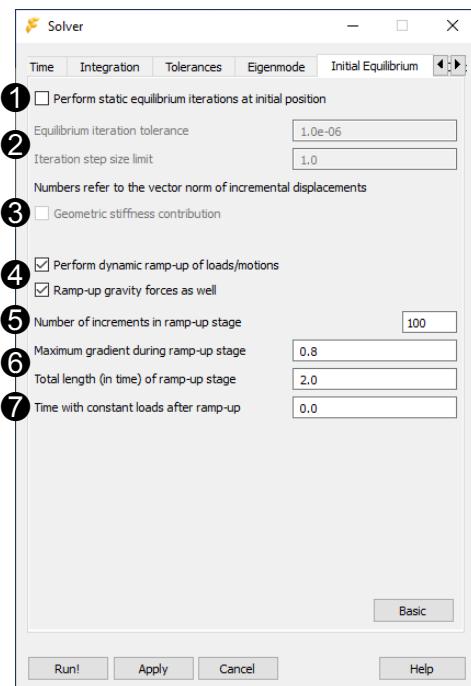


**CAUTION:** Computation of damped eigenmodes takes considerably longer time than computing the undamped modes.

### Initial Equilibrium tab

If, during modeling, your model is not positioned at its static equilibrium position, it is recommended that you perform the *Initial Equilibrium* analysis to move the mechanism to a resting position before simulating the dynamics.

- ① You can enable this option to perform static equilibrium iterations on the modeling configuration.
- ② You can adjust the iteration tolerance and the step-size factor for the initial static equilibrium iterations.
- ③ You can enable/disable the *Geometric Stiffness Contribution* in initial static equilibrium iterations (see the similar option in the [Integration tab](#)).
- ④ You can enable a dynamic load ramp-up simulation, optionally also with gravity force ramping. This can be used as an alternative (or supplement) to the initial static equilibrium analysis, if the initial loads are too large to be applied in a single step. See [Section 8.5.4, "Dynamic ramp-up of loads and prescribed motions"](#) for more information on this.
- ⑤ You can specify the number of time steps of the ramp-up simulation.
- ⑥ You can specify the maximum gradient of the scaling function and the total length (in time) of the ramp-up simulation stage. The time increment size of the ramp-up stage will then be set equal to the total length divided by the specified number of time steps.
- ⑦ You can specify a time for which the loads will be kept constant after completed ramp-up stage, before the actual load history is applied. This can be used to allow the state approaching a steady state with smaller velocities/accelerations, before starting the real dynamics.



The *Equilibrium iteration tolerance* is the convergence criterion on the norm of the iterative displacement correction during the initial static equilibrium iterations (equivalent to the *Scaled vector norm* tolerance for *Displacement iteration correction* in the [Tolerances tab](#)).

The *Iteration step size limit* is an upper limit on the norm of the displacement correction vector within one iteration. If the norm is higher than this value, the correction vector is scaled down such that its norm equals this number. Each time this happens, the iteration counter that is compared to the *Maximum* and *Minimum Number of Iterations* parameters of the [Integration tab](#), is reset to zero.



**NOTE:** Maximum and Minimum Number of Iterations set in the [Integration tab](#) also apply in the Initial Equilibrium analysis.



**TIP:** The defaults for the Equilibrium Iteration Tolerance and Iteration Step-Size Factor are usually acceptable. However, if the mechanism is modeled far from the equilibrium position, reducing the Iteration step-size limit may improve performance.



**CAUTION:** The Iteration step size limit must always be larger than the Equilibrium Iteration Tolerance.



**CAUTION:** To perform the initial equilibrium analysis, you may have to apply Additional Boundary Conditions before starting the simulation. (For information about applying such boundary conditions, see [Section 5.5.3, "Triad properties"](#)).

## Output tab

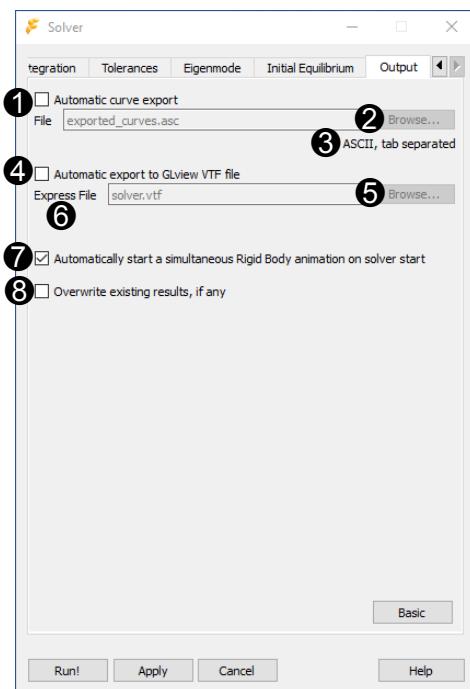
On this tab you can control the automatic curve and animation export from the Dynamics Solver to file. The automatic curve export is useful if you want to run Fedem in an iterative loop with some external software, and need to process selected solver output in order to calculate new input for subsequent runs. The automatic animation export writes a GLview VTF-file with the rigid body motion of the computed response. This file may then be opened in the Ceetron GLview software for further viewing (see [www.ceetron.com](http://www.ceetron.com) for further details on GLview).

- ① This toggle enables export of all curves in the model with **Export curve automatically** toggled on in the Property Editor panel (see [Section 9.2.4, "Curve properties"](#)).

- ② This field shows path and name of the file the curve data will be written to. Press **Browse...** to change file name or file format.

- ③ This label shows the selected format for the curve export. Available formats are MTS RPC (UNIX or PC formatting), and tab-separated multi-column ASCII.

Only curves plotting results produced by the Dynamics Solver can be exported in this manner. If you need to include curves with results from other Fedem solvers in the same output file (e.g. results from subsequent strain rosette analyses), you have to run the Fedem Curve Export Utility module instead (see [Section 8.13, "Automated curve export from multiple result database files"](#)).



- ④ This toggle enables export of the rigid body motion of the computed response to a GLview VTF-file.
- ⑤ This field shows path and name of the VTF-file that will be written. Press the **Browse...** button to change file name or file format.
- ⑥ This label shows the selected VTF file format. Available formats are Express, Binary and ASCII.
- ⑦ You can enable/disable the automatic start of a simultaneous rigid body animation when the Dynamics Solver is started. This is useful when doing rapid prototyping simulations of short duration, when it is essential to get quick feedback on the dynamic response.
- ⑧ You can enable/disable *Overwrite existing results, if any*, such that you don't have to delete the existing results before starting the Dynamics Solver. Instead, the result files are overwritten and no incrementation of the results database is performed. This means that the old results are lost, even if you close the model without saving after a simulation.



**CAUTION:** When the automatic VTF export is enabled, the FE data for all parts in the model is written to the specified VTF file before the Dynamics Solver is started. For large models this may take some time (especially if the FE data is unloaded), and the Fedem UI is blocked while the FE data is being exported.

### 8.5.3 Frequency response analysis

In some cases there might be input loads (or prescribed motions) in a mechanism which have variations over the time span of the simulation that makes it inconvenient to represent them in a direct time integration. It would require a very small time step size to resolve the load variation properly. In such cases the loads are better accounted for by considering them in the frequency domain instead of the time domain.

This is done by performing a *Fast Fourier Transform (FFT)* of each load function, resulting in a frequency-dependent load magnitude function, instead of a time-dependent function. The dynamics equation of motion is then transformed into the frequency domain and solved for a suitable range of driving frequencies. The frequency-dependent response can then be transformed back into the time domain through an inverse FFT, and optionally added to the response of other time-dependent loads. See the Fedem 7.6 Theory Guide, *Section 7.9, "Frequency Response Analysis"* for the theoretical foundation of the frequency response analyses.

A Frequency response analysis is activated in Fedem through a toggle in the basic Dynamics Solver Setup dialog box, see [Section 8.5.1, "Dynamics Solver \(Basic Mode\)"](#). In addition, you may need to specify some additional options for the Dynamics Solver (see [Section 8.2.3, "Additional solver options"](#)) in order to control the frequency response analysis. These solver options are listed in the table below.

<code>-sample_freq</code>	Sampling frequency [Hz] of the input loads to be used in the FFT calculation. The default value is 100 Hz.
<code>-windowSize</code>	Number of samples in each window in a segmented analysis. The default value is zero, meaning the entire time domain of the analysis is considered a single window (no segmenting).
<code>-nrModes</code>	Number of eigenmodes to use in a modal frequency response analysis. The default value is zero, which means a direct frequency response analysis is used.

The results of the frequency response analysis is stored on a separate `frs`-file in the results database, typically named `fd_p_#.frs`, where the `#`-character represents a number. You may then plot the response from the frequency response analysis by activating this file in the results file browser (see [Section 10.2.2, "Result manipulation"](#)), while deactivating the other `frs`-files generated by the Dynamics Solver.

### 8.5.4 Dynamic ramp-up of loads and prescribed motions

Sometimes you may want to start the dynamics simulation of a mechanism from a certain load or motion stage which is far from the modeling configuration. Ideally, you would like to apply this load/motion state by activating the *static equilibrium iterations* toggle in the "[Initial Equilibrium tab](#)", but this may fail due to that the loads are too large to be applied in one go, and/or the mechanism is singular such that a static analysis is not possible.

The *Perform dynamic ramp-up of loads/motions* option is a tool which may be used instead in such cases. It will perform a dynamics (or quasi-static) simulation over a specified time range (from  $t = t0 - T$  to  $t = t0$ , where  $t0$  is the start time specified in the "[Time tab](#)", and  $T$  is the total length of the ramp-up stage). In this simulation stage, all external loads and prescribed motions in the model that are functions of time, will be evaluated at  $t = t0$  and then scaled with a time-dependent smooth ramp function to obtain the load level that should be applied at a certain time. Optionally, the gravity loads may be scaled with the ramp function as well (although this is usually not necessary).

The scaling function that is used in the ramp-up simulation is a "[Smooth trajectory](#)" function (see [Section 5.11.2, "Function properties"](#)), which is defined by the following three parameters:

- Start: Equals  $t0 - T$  where  $t0$  is the specified simulation start time
- Length: The specified total length  $T$
- Max ( $f'$ ): Equals the specified maximum gradient

The fourth parameter,  $\text{Max}(f'')$ , will then be computed automatically such that the function value at  $t = t0$  equals one. This puts a constraint on the allowable values on the *Length* and *Max (f')* parameters; they need both to be larger than zero, and the product *Length*\**Max(f')* has to be greater than one but not greater than two.



**NOTE:** Since the ramp-up simulation is performed for times less than the specified "Start time" (which usually is zero), there will be no response output from this stage by default. Therefore, if you do want to save the results also for this stage, you have to specify a "[Output start-up time](#)" less or equal to  $t0 - T$  using the "-savestart" solver option (see [Section 8.5.5, "Result output control"](#) below).

## 8.5.5 Result output control

To control the size and contents of a dynamics analysis result database, several additional options exist.

The results data from a dynamics analysis is divided into primary and secondary variables. The primary ones consist of triad- and superelement (part) position matrices, and, if any, generalized displacements for super-elements having component modes. All other variables are secondary variables. The output frequency, accuracy and the amount of these variables may be controlled using Additional Solver Options for the Dynamics Solver (see [Section 8.2.3, "Additional solver options"](#)). These options are discussed below. Refer to [Appendix C, "Command line options"](#) for a complete list of solver options.

### Output start-up time

Before presenting the options to format the output, we note that the time of output start-up may also be controlled. This is done using the “-savestart” solver option. The option applies to both the primary and secondary variables, as well as internal control system data.

-savestart Time for first save to response database



**NOTE:** The default value of this option is 0.0. Therefore, if you are using a “Start time” less than zero and want to save the results also for the time steps with physical time less than zero, you have to specify “-savestart <start time>” as an additional solver option for the Dynamics Solver.

8

### Output frequency

Primary variables are output for all time steps of a dynamics simulation and, while using default settings, so are the secondary ones. The output frequency of secondary variables, however, may be lowered by using the “-saveinc2” solver option.

-saveinc2 Time between each save of secondary variables

For models with an internal control system, the “-saveinc2” option applies by default to control system data that is needed in a restart simulation as well (see [“Time tab” in Section 8.5.2](#)). However, it is possible to specify a separate output frequency for those data through the option “-saveinc4”

-saveinc4 Time between each save of control system data

### Output accuracy

Primary variables are by default output in double precision (64-bit real). However, the output precision may be set to single by specifying the solver option “-double1-” (the appending minus sign indicates a *false* setting). This will make the primary results file created by the Dynamics Solver half the normal in size.

-double1- Save primary variables in single precision



**CAUTION:** You should not switch off double precision output for primary variables unless you are particularly low on disk space, as this may affect the accuracy of the subsequent recovery runs. This is particularly true for models experiencing large global displacements but small local deformations.

Secondary variables are by default output in single precision (32-bit real). This is sufficient for most purposes, and contributes to keeping the disk space needed by the secondary results file at a low level in long simulations. However, if you plan to use some of secondary variable results as input functions in subsequent simulations, it can be essential to retain full precision in the saved results to obtain satisfactory accuracy. For this purpose, the solver option “-double2” can be used.

-double2 Save secondary variables in double precision

When specified, most of the secondary variables will be saved in double precision. Some quantities that are not likely to serve as input in subsequent runs are always saved in single precision. Thus, the “-double2” option currently affects only the following types of quantities:

- Triad velocities, accelerations and forces
- Spring stiffnesses, lengths, deflections and forces
- Damper coefficients, lengths, velocities and forces
- Friction forces
- External force values and vector components
- Joint variables, velocities and accelerations
- Tire contact point and wheel carrier forces
- Control line variables

## Output selection

While using default settings, all primary variables but no secondary available are output to file. The selection of secondary variables to output may be customized in different ways. The “`-allSecondaryVars`” option can be used to switch on the output of *all* secondary variables available. In addition, a set of options of the form “`-all<NameOfEntity>Vars`” are used to control the result output for a range of entities. (E.g. “`-allTriadVars`” requests the output of all variables related to triads, while “`-allForceVars`” requests the output of all force variables in the simulation.)

All available solver options are listed in [Section C.3, "Dynamics solver options \(fedem\\_solver\)".](#)

It is also possible to toggle output of selected results for individual objects. This is done through a set of toggles in the Property Editor panel for some object types (Triads and Joints). For other object types, it can be done by entering commands in the *Description* field of the objects for which output is requested. All these commands are given in [Appendix D, "Beta feature documentation".](#)

## File buffering and flush frequency

As stated above, the dynamics solver writes the results to (up to) three different results files. Since the amount of data output per time step and output frequency are different for these files, it is important to be able to synchronize the actual data output at given time steps. This is needed especially when doing a simultaneous simulation and animation/curve plotting. For this purpose, the “`-flushinc`” option can be used.

<code>-flushinc</code>	Time between each database file flush
< 0.0 :	Do not flush results database (let the OS decide)
= 0.0 :	Flush at each time step, no external buffers (default)
> 0.0 :	Flush at specified time interval, use external buffers

The default action (`flushinc = 0.0`) will flush all open results files at the end of each time step. Each file is associated with an internal file buffer of fixed length. For small models the amount of data per time step is usually less than the size of this buffer. The data will therefore not be written to disk at the end of the time step, unless we force an explicit flush.

Instead of relying on the fixed-size internal buffer, you may also instruct the solver to do an explicit flush at fixed time intervals (`flushinc > 0.0`). An external buffer is then allocated for each result file which is big enough to just hold the number of time steps corresponding to the given flush

interval. This can be used to reduce the number of file IO operations to a minimum, and may speed up the overall solution time if you have a slow disk (or data network if using a remote device). Moreover, if the model is so big that a single time step does not fit in the internal buffer<sup>1</sup>, it will write data physically to the file in several steps during a time step, and you may risk synchronization problems in a simultaneous simulation animation/plotting. Specifying a flushinc greater than or equal the simulation time step size will prevent this.

By specifying a flush interval less than zero, you let the operating system decide when it wants to physically write data to file. When running the solver batch, or through the user interface with no graphs or animations loaded, this is normally sufficient (unless you have a slow disk drive, then flushinc > 0.0 may work better).



**TIP:** *For small models (i.e., few system DOFs) that run over a long time span using a huge number of time steps, the default value (flushinc = 0.0) may severely hamper the performance, as it is forced to write small data amounts to disk at a very high frequency. In such cases, using a value less than zero or (much) greater than the time step size is recommended.*

### 8.5.6 Stress and strain recovery during time integration

It is possible to perform recovery of internal displacements and stresses (von Mises stress only) for FE parts during the dynamics simulation itself, instead of doing it as separate processes after the dynamics simulation process is finished as described in [Section 8.1, "Stress recovery"](#). Similarly, strain rosette recovery can also be performed during the dynamics simulation. This has the great advantage that you can monitor these results while the dynamics simulation is still running.

To activate recovery during simulation, use the appropriate toggles in the ["Advanced tab"](#) of the Property Editor panel for the Part in question (shown at right, see also [Section 5.1.5, "Part properties"](#)). You may enable one or both of these toggles to activate such recovery during simulation for an FE Part, and can also enable them for several parts in a multi-part model. Enabling strain rosette recovery will, of course, have no effect unless strain rosette objects have been defined on that FE part.

- |   |
|---|
| <input type="checkbox"/> Perform stress recovery during dynamics simulation         |
| <input type="checkbox"/> Perform strain rosette recovery during dynamics simulation |

1. The size of the internal file buffer is platform dependent. Consult the technical documentation on your computer operating system to find out how big it is.



**CAUTION:** Doing stress recovery during the dynamics simulation is a costly operation, so for models with large FE parts, this will increase the total simulation time considerably. Therefore, use this feature only if the simulation time is not essential.

### 8.5.7 Monitoring the most problematic DOFs during time integration

For complex models, it is not likely that the dynamics simulation will run smoothly in the first attempt. Some fine tuning and/or model correction will often be required before a converged solution can be obtained. To aid the debugging of problematic models, the DOFs that get the largest solution increments during the non-linear iterations will be listed in the `.res` file when the simulation experiences convergence problems. Often, the problem can then be identified by studying the elements related to these DOFs and verifying that their geometry and/or properties are sane.

Fedem will produce those messages when the convergence criteria employed (see "[Tolerances tab](#)" in [Section 8.5.2](#)) increases in two or more consecutive iterations, or the number of iterations is getting close to the maximum number of iterations specified. It is then assumed that a convergence problem is encountered, and the specified number of the worst DOFs will then be output along with local DOF number, object entity information, and the actual solution increment in that DOF.

Finally, you can specify the Dynamics Solver to abort the simulation after a certain number of such poor convergence warnings have been issued.

The DOF monitoring is controlled by means of Additional Solver Options (see [Section 8.2.3, "Additional solver options"](#)). The following options are available for this purpose (default value in parentheses):

- monitorWorst*      Number of DOFs to monitor on poor convergence (6)
- monitorIter*        Number of iterations to monitor before *maxit* (2)
- stopOnDivergence* Number of warnings on possible divergence before the dynamics simulation is aborted (0 = no limit)

### 8.5.8 Starting the analysis



Once you have set up the dynamics solver options, you can start the dynamics simulation by clicking the **Solve Dynamics** button on the *Solvers* tool bar or in the *Solve* menu.

### 8.5.9 Handling singularities during the dynamics analysis

The Dynamics Solver has a similar treatment of singular equations during the solution of linear equation systems, as the FE model Reducer (see [Section 8.3.4, "Handling singularities during the model reduction"](#)). All singular DOFs will thus be attempted found in one go, letting you fix them all before a new analysis is attempted. However, in the Dynamics Solver, neither type of singularities (true zero pivots, or reduced-to-zero pivots) are permitted. The solver will therefore abort on any occurrence of singularities after the triangularization is completed. All singularities found are then listed in the *Output List* and on the *.res* file. They are identified with the internal node number and the mechanism entity it is associated with (Triad DOF, Joint DOF or component mode of a Part).

## 8.6 Stress recovery analysis

### 8.6.1 Stress recovery options

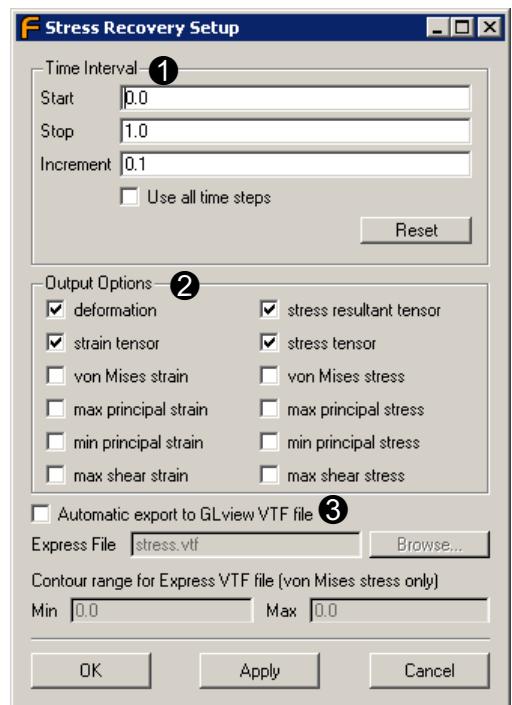


To specify parameters for the stress analysis, click the **Stress Recovery Setup** button on the *Solvers* tool bar (or *Solve* menu).

Each time stress recovery is run, the results are added to the existing stress recovery results. This means you could solve stresses for one time interval first, and subsequently for another interval, while stresses from both intervals could be animated in the same animation. You could also solve stress first, view them, and solve strains later, making both stress and strain available for post-processing.

This also means that if stress recovery is performed more than once using identical settings, the same results will be stored multiple times on disk. It is not checked whether the results you asked for already exist or not.

- ① To specify the time steps at which stress is recovered, a *Start* time, *Stop* time and a time *Increment* should be provided. However, if the **Use all time steps** option is enabled, stress recovery will be performed for all computed time steps between the specified start and stop time. The **Reset** button restores the default *Time Interval* values, which are equal to the start and stop times of the simulation as specified in the *Time tab* of the Dynamics Solver Setup dialog box, and 10 times the (initial) time increment used in the simulation.



- ② You can choose which type of results to recover by activating the appropriate toggles. For very large models it is recommended not to recover more results than you actually want to animate, as the efficiency of the animation process, and the size of the results database files, is dependent on the amount of data computed.



**NOTE:** If you choose to recover the stress and/or strain tensors, there is no need to also toggle on the derived quantities (von Mises, max principal, etc.). You may always animate the derived quantities as long you have recovered the stress/strain tensors using the Operation menu in the Property Editor panel of the animation (see "[Contours tab](#)" in [Section 9.5.2](#)).

- ③ You may enable direct export of a GLview Express VTF-file with von Mises stress contours and optionally the deformed shape, for further viewing in the Ceetron GLview environment (see [www.ceetron.com](http://www.ceetron.com)). You may also need to specify the contour range to be used in the exported VTF-file (the max and min. values of the exported von Mises stress will be used if no range is specified).



**NOTE:** The deformation toggle in the Output Options frame also affects the GLview VTF export from the Stress Recovery. The deformed shape will thus be exported to the VTF file only when this toggle is on. None of the other toggles influence the VTF export.

## 8.6.2 Result output control

As explained above, the Stress Recovery Setup panel may be used to specify the result types wanted as output from the stress analysis. There are, however, some additional options for customizing the size and contents of the results database that may be specified in the Additional Solver Options dialog box (see [Section 8.2.3, "Additional solver options"](#)). Some of these options are discussed below. The complete list of options to the Stress Recovery solver module is found in [Section C.4, "Stress recovery options \(fedem\\_stress\)"](#).

### Output accuracy

By default the results from a stress recovery analysis are output in single precision (32-bit real). The output precision may, if so wanted, be set to double precision (64-bit real) by using the “-double” solver option.

## 8.6.3 Starting the analysis



Once you have set up the stress recovery options and performed the dynamics simulation, you can start the stress recovery by clicking the **Recover Stress** button on the *Solvers* tool bar (or *Solve* menu). You may also run the stress recovery on a selection of parts or on individual element groups, see [Section 8.2.5, "Part- and group-wise solving"](#).



**NOTE:** If element calculations fail during stress recovery analysis, the recovery continues on the other elements. All failed elements will appear gray in the contour plot after the animation is loaded.



# 8.7 Mode shape recovery analysis

## 8.7.1 Mode shape options

To animate or display detailed mechanism mode shapes, you must first set up and perform the dynamics analysis and then the mode shape recovery analysis.

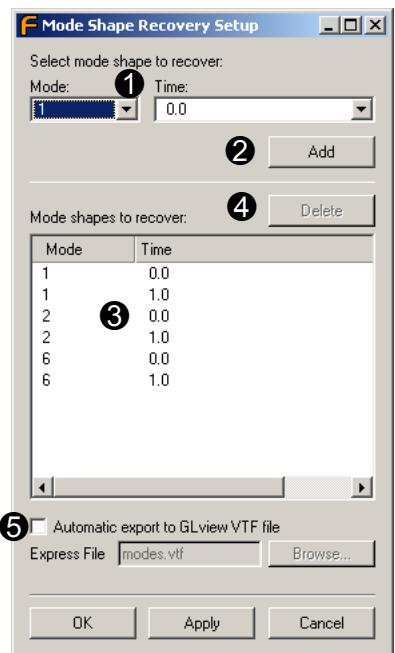


**NOTE:** Rigid body mode shapes can be animated without a mode shape recovery analysis (see [Section 9.5.2, "Animation properties"](#) for more information).



To set up the mode shape analysis, click the **Mode Shape Recovery Setup** button on the *Solvers* tool bar (or *Solve* menu). The dialog box is opened as shown below. You can then select the modes you want to expand and add them to the list of selected modal results.

- ① *Mode and Time* – pull-down menus with mode numbers and times at which eigenmodes have been (or will be) calculated in the dynamics analysis.
- ② **Add** button – inserts selected mode shape for the selected time into the results list.
- ③ *Mode shapes to recover* – list of selected mode shapes to expand during postprocessing (sorted by mode number).
- ④ **Delete** button – removes the selected mode and time from the results list.
- ⑤ You may enable a direct export of a GLview Express VTF-file for each recovered mode shape, for further viewing in the Ceetron GLview environment ([www.ceetron.com](http://www.ceetron.com)). If a mode shape is to be recovered for more than one time step, that shape will be exported to VTF only for the first time step specified.



**TIP:** The Mode and Time pull-down menus both have an (All) entry in the bottom to facilitate easy selection of all entries in the list. This is useful if you want to recover all mode shapes and/or the selected mode shape for all time steps at which it has been computed.



**NOTE:** The expanded mode shapes for one part are stored in a single frs-file provided all modes are expanded for the same set of time steps. However, if only one mode is expanded for a different set of time steps than the other modes, one frs-file is created for each expanded mode. Therefore, if you are expanding a large amount of modes, it is advised to select the same time steps for all modes in order to limit the number of files in the results database, as this might impact the post-processing performance.

## 8.7.2 Starting the analysis

Once you have set up the mode shape recovery options and performed the dynamics simulation, you can post process the modes at the selected times by clicking the **Recover Mode Shapes** button on the *Solvers* tool bar (or *Solve* menu).

When Fedem has postprocessed all the selected modes, you can animate the mode shapes (see [Section 9.5, "Animations"](#) for more information).

## 8.8 Strain rosette analysis

The strain rosette analysis recovers the stresses and strains on the virtual strain rosettes defined in your model. See section [Section 5.12.3, "Strain rosettes"](#) on how to create and edit virtual strain rosettes. The output is similar to the output from real strain gages in addition to standard strain and stress quantities like Von Mises, principal stresses/strains, and angle of max/min. principals.

### 8.8.1 Strain rosette options



To specify parameters for the strain rosette analysis, click the **Strain Rosette Recovery Setup** button on the *Solvers* tool bar (or *Solve* menu).

- ➊ You can specify the *Start* and *Stop Time* for strain rosette recovery and the *Time Increment* at which to recover the rosettes. If the *Use all time steps* option is enabled, rosette recovery will be performed for all computed time steps between the specified start and stop time. The **Reset** button restores the default *Time Interval* values, which are equal to the start and stop times of the simulation as specified in the *Dynamics Solver Setup*, and using all time steps in between.
- ➋ You may enable a direct export of gage strains to DAC files with a specified sample rate.
- ➌ You may enable a stress cycle count with a specified bin size. The results are written to the file *fedem\_gage.res* when running the strain rosette recovery. The computed damage will also be reported to this file for all strain rosettes when this option is enabled.
- ➍ Strain rosette definitions can be imported from a file by pressing this button. The strain rosettes defined will then be read into Fedem, and virtual strain rosettes will be created automatically based on the definitions. See [Section 8.8.4, "Strain rosette definition file format"](#) for the format of this file.



## 8.8.2 Starting the analysis



Once you have set up the strain rosette recovery options and performed the dynamics simulation, you can start the strain rosette recovery by clicking the **Recover Strain Rosettes** button on the *Solvers* tool bar (or in the *Solve* menu).

## 8.8.3 Result output

Results for all strain rosettes on a part are output to the binary `.frs` file named `<partname>_#.frs`. The `.frs` file allows the post-processing of strain rosette recovery results through graphs.

In addition to the stress and strain results, you may also output the nodal deformations in the strain rosettes to the `.frs` file by specifying `-deformation` as an additional option to the Strain Rosette Recovery (see [Section 8.2.3, "Additional solver options"](#)).

If enabled, a file named `rosette<ID>_gage<n>.dac` is written for each strain gage, containing the strain of leg `<n>` in strain rosette `<ID>`. This file is output in the nCode DAC format.

It is also possible to output the strain rosette result directly to ASCII files. By specifying `-writeAsciiFiles` as an additional option to the Strain Rosette Recovery, you will get a file named `rosette<ID>.asc` for each strain rosette defined. A summary of contents for this ASCII file is given below.

All of the above mentioned files are created in separate directories for each part, which then are placed in the sub-directory `timehist_gage_rcy_####/` in the result file hierarchy (see [Section 10.3, "RDB directory structure"](#)).

### ASCII output file format

The `rosette<ID>.asc` output file contains a heading and a data section. (See file cutout below.) The heading summarizes the strain rosette definition, the data section lists all measurements made.

```

# Strain rosette identifier:           1      Link :      1
# Global nodes          :     844      846      830      828
# Number of gages       :      2
# Angle between gages   :    90.000
# Position              :  2.336E-001 -7.206E-001 4.500E-002
# Position along Z-axis  :  0.000E+000
# X-direction unit vector: 0.000E+000 1.000E+000 0.000E+000
# Y-direction unit vector: -1.000E+000 0.000E+000 0.000E+000
# Z-direction unit vector: 0.000E+000 0.000E+000 1.000E+000
#
#          time      eps_x      eps_y      gamma_xy      eps_gage1      eps_gage2
0.000000E+000 -9.751E-018 -3.048E-018 2.205E-018 -9.751E-018 -3.048E-018
2.000000E-003 1.835E-006 4.353E-007 -2.367E-007 1.835E-006 4.353E-007
4.000000E-003 6.391E-006 1.490E-006 3.337E-007 6.391E-006 1.490E-006
6.000000E-003 1.041E-005 2.397E-006 1.660E-006 1.041E-005 2.397E-006
8.000000E-003 1.008E-005 2.320E-006 1.621E-006 1.008E-005 2.320E-006

```

Note that the cutout above is an edited version not showing the entire file contents. Detailed information on the full contents of the data section is listed below.

Data output	Description of output
Time	Time
$\varepsilon_x$	Strain XX in rosette coordinate system
$\varepsilon_y$	Strain YY in rosette coordinate system
$\gamma_{xy}$	Shear strain XY in rosette coordinate system
$\varepsilon_1$	First principle strain
$\varepsilon_2$	Second principle strain
$\gamma_{max}$	Max shear strain
$\alpha_1$	Angle from rosette X-axis to direction of principle strain
$\alpha\gamma$	Angle from rosette X-axis to direction of max shear
$\sigma_x$	Stress XX in rosette coordinate system
$\sigma_y$	Stress YY in rosette coordinate system
$\tau_{xy}$	Shear stress XY in rosette coordinate system
von Mises	von Mises stress
$\sigma_1$	First principle stress
$\sigma_2$	Second principle stress

Data output	Description of output
$\tau_{\max}$	Max shear stress
$\varepsilon_{\text{gage}1}$	Strain in leg1
$\varepsilon_{\text{gage}2}$	Strain in leg2
$\varepsilon_{\text{gage}3}$	Strain in leg3

All angles are output in degrees to the ASCII file.

### File buffering and flush frequency

The output to the frs-file may be controlled using the additional option *-flushinc*, which works in the same manner as with the Dynamics Solver (see [Section 8.5.5, "Result output control"](#)). However, the default value is here -1.0, i.e. the file is flushed to disk when the internal fixed-size file buffer is filled. Note that this option only affects the frs-file output. The DAC and ASCII file output is not affected.

#### 8.8.4 Strain rosette definition file format

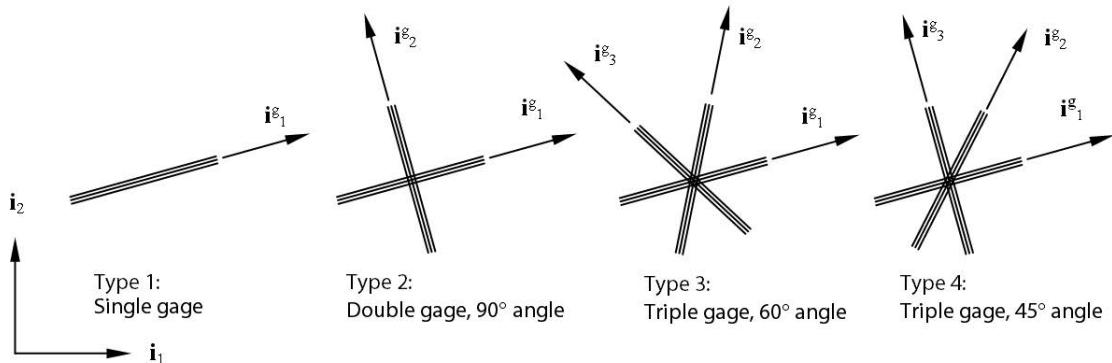
It is possible to define strain rosettes in an ASCII input file, as shown in the example below, and read them into Fedem.

```
#id type link numNodes   n1   n2   n3   n4   zPos   X_x   X_y   X_z   Z_x   Z_y   Z_z   E-mod   nu
10   2       1       4     844  846  830  828   0.0    0.0   1.0   0.0   0.0   0.0   1.0   2.06e+11  0.0
20   4       1       3     845  846  828           0.01   0.0   1.0   0.0   0.0   0.0   1.0   2.06e+11  0.0
end
```

Each line defines a strain rosette, where the following data must be given:

**id:** The id number is used for naming purposes of the result files. The above example defines two strain rosettes, with identifiers 10 and 20 respectively.

**type:** According to the figure below, rosette 10 is of type 2, i.e. a two gages with 90° degrees between the gages, whereas rosette 20 is of type 3, three gages with 60° degrees between the gages.



**link:** The superelement (part) number that the present rosette is attached to.

**numNodes n1 n2 n3 (n4):** Number of nodes for the element followed by node numbers that define the element topology. Permissible values for numNodes are 3 and 4, giving CST triangle and bi-linear quad element respectively. The nodal numbers must be given in a circular fashion around the element, where both clockwise and counterclockwise sequences are permitted. The node numbers will possibly be reversed internally so that the element Z-axis points in the user defined direction.

**zPos:** defines the location of the rosette along the Z-axis. If the user wants the strains measured at the top of a plate, the number at the Z-pos column should be  $h/2$  where  $h$  is the plate thickness. A value of  $-h/2$  will give the strains at the bottom of the plate.

**x\_x x\_y x\_z:** These three numbers define a vector, which is used to define the local X-direction for the rosette coordinate system. The vector is given in the part coordinate system. The vector does not need to be given as a unit vector, nor does the vector need to lie exactly in the element plane. The X-direction used for computations is obtained through a projection of the user-defined vector onto the element plane.

**z\_x z\_y z\_z:** These three numbers represent a vector, which is used for defining the positive Z direction of the rosette coordinate system. The vector defined by the user does not need to be exactly perpendicular to the element plane. As long as the vector has a component pointing out from the element plane, the

Z-direction for the rosette coordinate system will be correctly defined. The rosette Y-direction is implicitly defined through the X- and Z-directions.

**E-mod nu:** Young's modulus (E-mod) and Poisson's ratio (nu) are used for computing the stress state at the rosette location.

## 8.9 Strain coat analysis

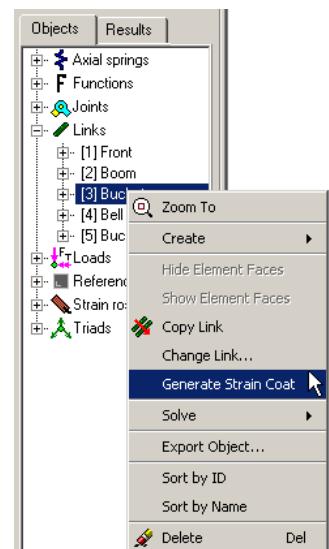
This analysis recovers the stresses and strains on all strain coat elements in the model, and calculates a summary of the recovered results as it processes. The output from the strain coat analysis is a result database file (.frs file) for each part containing the maximums of certain stress/strain quantities over the time interval considered. Optionally, you may also perform a rainflow and fatigue analysis based on the computed stress or strain histories during the strain coat recovery. The result files from the strain coat analysis are created in separate directories for each part below the *summary\_rcy\_####/* directory in the result file hierarchy (see [Section 10.3, "RDB directory structure"](#)).

### 8.9.1 Generating strain coat

Before you run the strain coat analysis, you must generate strain coat elements for the parts or element groups in question. This is done by right-clicking the parts or groups in the *Objects* list of the Model Manager panel and selecting **Generate Strain Coat** from the menu. Again, multi selection is possible.

One strain coat element is created for each non-interior face of the finite elements in the current selection. If a strain coat element already exist for a given face, a new strain coat element is not created. Therefore, repeating the **Generate Strain Coat** command for the same selection has no effect.

When creating strain coat elements by selecting an element group the new strain coat elements are automatically added to that group.



If a part is selected, the created strain coat elements are only added to the implicit element groups which the parent finite element belongs to, not to any explicit element groups that the parent element might be a member of. Refer to [Section 5.2, "Element groups"](#) to learn about implicit and explicit groups in Fedem.



**NOTE:** The strain coat elements will appear in the .ftl file in the FE model repository when the model is saved.

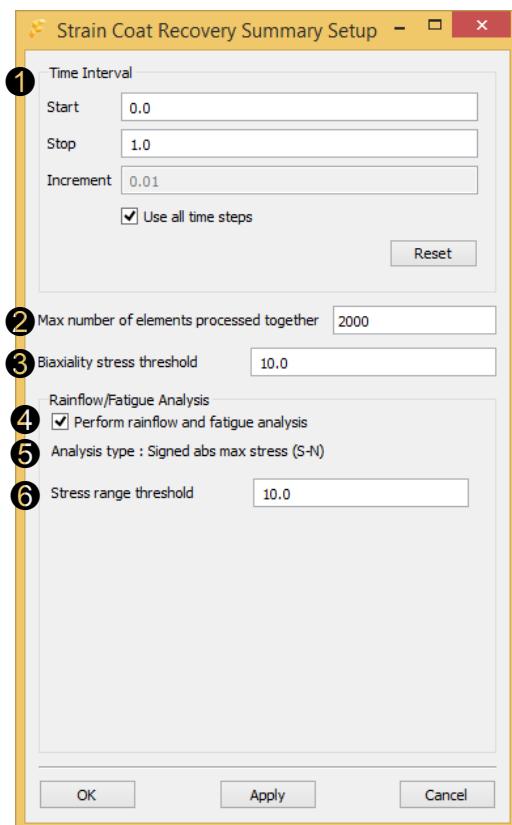
## 8.9.2 Strain coat analysis options



To specify parameters for the strain coat analysis, click the **Strain Coat Recovery Summary Setup** button on the **Solvers** tool bar (or **Solve** menu).

① You can specify the *Start* and *Stop Time* for strain coat recovery summary and the *Time Increment* to be used. However, if the **Use all time steps** option is enabled, all computed time steps between the specified start and stop time will be considered. The **Reset** button restores the default Time Interval values which are equal to the start and stop times of the simulation as specified in the Dynamics Solver Setup dialog box, and to use all time steps in between.

② You may limit the number of elements to be processed concurrently by adjusting this value. Especially for large parts and long time series this might be necessary due to higher memory requirements.



- ③ You may set a gate value for the *Biaxiality* calculation. That is, the mean biaxiality will be computed only for elements whose max principal stress is larger than the specified threshold value.
- ④ You may toggle on/off rainflow and fatigue analysis. The remaining options in this dialog box are sensitive only when this toggle is on.
- ⑤ The Analysis type is always *Signed abs max stress (S-N)*.
- ⑥ *Stress/Strain range threshold*: You may set a gate value for the *Peak valley extraction*. That is, only the stress/strain ranges with magnitude higher than this value will be included in the peak valley extraction.

To perform rainflow and fatigue analysis during the strain coat recovery, you also need (unless nCode is used) to assign an S-N curve to base the damage calculation on for each element group you want to consider in the fatigue analysis. This is performed in the Property Editor panel for the element groups, see [Section 5.2.1, "Element group properties"](#).

### 8.9.3 Starting the analysis



Once you have set up the strain coat recovery summary options and performed the dynamics simulation, you can start the strain coat recovery by clicking the **Recover Strain Coat Summary** button on the **Solvers** tool bar (or **Solve** menu). The Strain Coat Recovery is then performed on each part in the model containing strain coat elements. Parts without strain coat elements are automatically omitted.

### 8.9.4 Strain coat recovery on element groups or individual parts

A strain coat recovery can also be performed on defined element groups or a selection of parts. See [Section 8.2.5, "Part- and group-wise solving"](#) to learn how such a process is started. In this case, strain coat elements are created automatically for the selected part(s) or element group(s), before the recovery process is started. It is therefore no need to do this manually, as explained in [Section 8.9.1, "Generating strain coat"](#), in this case.

## 8.10 Direct linear analysis of FE Parts

Instead of doing a full dynamic simulation with subsequent stress recovery, it is also possible to perform a direct linear FE analysis on individual parts. This is handy if you only want to asses the static properties of a FE model before using it in a larger mechanism simulation.

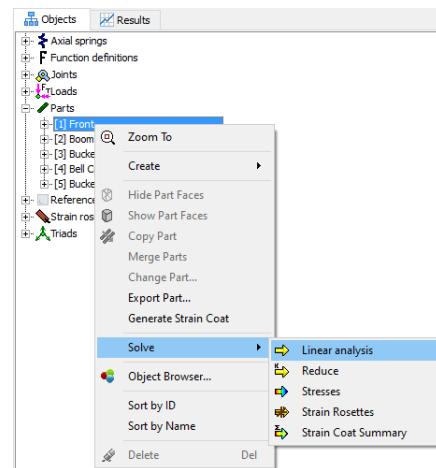
A direct linear FE analysis of a Part is performed while keeping all Triads defined on the part fixed. Alternatively (or in addition), more detailed boundary conditions on individual nodal points in the part may be defined in the FE model file (see [Section 1.5, "Using FE models"](#)). Any loads defined in that file will also be accounted for, in addition to gravity loads (if a gravitation vector is defined, see [Section 4.9, "Gravitation"](#)). All other mechanism objects (Joints, springs, etc.), if any, will be ignored. Therefore, performing a linear analysis on an assembly of FE parts will be the same result as performing the analysis on each part separately.

### 8.10.1 Starting a direct linear analysis

A direct linear analysis can only be started as a Part-wise solve process, see [Section 8.2, "Part- and group-wise solving"](#), since no coupling will exist between multiple parts in the model during the analysis.

Start the process by right-clicking on the part (or parts) in the *Objects* list of the Model Manager panel, and choose **Solve** and then **Linear Analysis** from the menu (as shown to the right).

The results of a linear analysis is only the displacement field, which then can be viewed in an Animation object. However, it is possible to also perform a Part-wise stress recovery on the same part afterwards, to compute stresses based on the computed displacement field. This stress recovery will perform the same simulation as the normal stress recovery process described in [Section 8.6, "Stress recovery analysis"](#), except that the recovered displacements are read from a file instead of being computed from the system response of the dynamics simulation.



**NOTE:** The result files from a Linear analysis is stored in the same location as the results files from a stress recovery of that part.

## 8.11 Interaction during processing

### 8.11.1 Simultaneous viewing and processing

If you already have set up graphs and animations (see [Chapter 9, "Postprocessing Results"](#)), they can be viewed during the dynamics analysis. Your graphs and 3D views can be dynamically updated to show the results from the mechanism as they are computed.

#### Graphs

If graph views are displayed (see [Section 9.2.2, "Showing a graph"](#)) during the dynamics simulation, each curve is continuously updated with values from the solution, reflecting the progress of the dynamics analysis.



**TIP:** A graph showing the number of iterations for each time step gives a good indication of both the progress of the simulation, and how quickly each time step converges to the specified tolerance.

#### Animations

If a 3D animation is loaded (see ["Loading animations" in Section 9.5.1](#)) during the dynamics simulation, the 3D view of the model is continuously updated with the rigid body motion of the part, reflecting the simulation results. The model can also be examined (rotated, zoomed, and so on) during the animation.



**TIP:** The rigid body animation is an effective and intuitive way to observe the progress of the simulation. When the rigid body animation is displayed, a progress bar also appears in the upper-right corner of the Modeler view.



**TIP:** A rigid body animation may be started automatically when the dynamics solver starts by enabling a toggle in the Dynamics Solver Setup (see ["Output tab" in Section 8.5.2](#)).



**TIP:** Animations can be loaded and closed at any time during the simulation.



**CAUTION:** With large models and long simulations, the simultaneous visualization and simulation features may use a large amount of system resources.

### 8.11.2 Stop processing



Solution processes can be stopped at any time by pressing the **Stop All Solvers** button on the **Solvers** tool bar (or **Solve** menu).



**NOTE:** All results created before the **Stop** button is pressed are stored and can be evaluated in the normal manner.

## 8.12 Deleting results



To delete result files from the dynamics simulation and recovery operations (stress recovery, modes recovery, strain rosette recovery and strain coat recovery), click the **Delete Results** button on the *Solvers* tool bar (or *Result* menu).



**NOTE:** Result files are not deleted until you save or close the model, and files from model reductions are never deleted.



**CAUTION:** If you use the **Save As...** command to save a model to a different location, result files from previous solutions are not copied to the new location. Only result files and reduced parts from the current analysis are copied. (Applicable only if the "Discard results" and "Discard reduced parts" toggles are not set)

### 8.12.1 Deleting specific results

Results from specific recovery processes can be deleted. This is useful for reclaiming disk space or changing the 'result window' available to the post processing.



The stress recovery results can be deleted by clicking the **Delete Stress Recovery results** button on in the *Results* menu.



The mode shape recovery results can be deleted by clicking the **Delete Mode Shape Recovery results** button on in the *Results* menu.



The strain rosette recovery results can be deleted by clicking the **Delete Strain Rosette results** button on in the *Results* menu.



The strain coat recovery results can be deleted by clicking the **Delete Strain Coat Recovery Summary results** button on in the *Results* menu.



**NOTE:** The results files are deleted immediately when performing these actions.

## 8.13 Automated curve export from multiple result database files

If you need to export curve data from several result database files into a single output file (e.g., if you want curves from one or more Strain Rosette analyses exported to a single RPC-file), that can be done by executing the Fedem Curve Export Utility module after the necessary solver tasks have

been completed. This module can only be invoked as a separate command from a terminal window or command prompt.

Start the curve export by issuing the following command:

```
fedem_graphexp -frsFile (fnames) -modelFile (mname) \
-curvePlotFile (cname) [<options>]
```

where (*fnames*) is a list of one or more frs-files on the form '*<file1, file2, ...>*', (*mname*) is the Fedem model file (*.fmm*) in which the curves to be exported are defined, and (*cname*) is the name of the output file where the curve data is exported. The module also have some other options to facilitate the export process (see [Section C.8, "Curve export options \(fedem\\_graphexp\)"](#) for a complete list of all options).

## 8.14 Batch execution of solver processes

If you need to run several versions of a model, it is often most efficient to edit the model file directly in an editor, or by means of scripting, and then run the solver processes in batch mode from a terminal window or the command-line prompt. To facilitate such batch executions, a set of command-line options is provided, that run Fedem in a non-graphics and non-interactive mode (see [Section C.1, "Fedem GUI options"](#) for the complete list of command-line options for the Fedem UI).

### 8.14.1 Batch solving trough the User Interface

Start the Fedem UI from the command-line prompt with the following command:

```
fedem -f (mname) -solve (sname)
```

where (*mname*) is the Fedem model file and (*sname*) is one of the keywords *reducer*, *dynamics*, *stress*, *modes*, *strainage*, *straincoat*, *events* or *all*. This will read the specified model file, and immediately launch the specified solver process(es), and then save the model file and the obtained results before the command exits. Thus, this has the same effect as (but is normally faster) the following steps:

- Start the Fedem UI interactively in the normal manner
- Open the desired model file
- Launch the desired solver process from the *Solvers* tool bar
- Exit Fedem with saving of model file

In the same way as when running from the *Solver* tool bar (or from the *Solve* menu), it is checked that all parts are reduced and up to date, and that dynamics results exist, before a recovery process is started. And if needed, the necessary pre-requisite solver tasks are executed first. Thus, to simply run all recovery types on all parts for a given model file, you can just use the option *-solve all* and all required solver tasks will be executed sequentially without user interaction.

### 8.14.2 Preparing for batch solving on remote computers

Sometimes it is desirable to execute solver processes directly on another computer than the Fedem UI is run on, and which have a separate file system. It is then necessary to create the needed RDB directory structure locally, populated with all the required solver input files, and then manually transfer it to the remote computer for batch execution (see [Section 10.3, "RDB directory structure"](#) for details on the RDB structure). When finished, the whole directory structure can then be copied back onto the local computer for further post-processing in the Fedem UI.



To create a complete RDB structure with input files for all the Fedem solver processes, select **Prepare for batch execution** in the *Solve* menu.

The RDB structure required for a certain solver process can also be created by issuing the following command from a terminal window:

```
fedem -f (mname) -prepareBatch (sname)
```

where the meaning of *(mname)* and *(sname)* is as explained above in [Section 8.14, "Batch solving trough the User Interface"](#). Thus, the effect of this command is exactly the same as with the *-solve* option, except that the solver processes themselves are not executed. Specifying *(sname)=all* with this command is equivalent to using the **Prepare for batch execution** entry in the *Solve* menu. When the command have exited you have the necessary RDB directory structure, which can be transferred to the remote computer for batch execution.

## 8.15 How to read error messages from the solvers

When a solver process fails to complete a simulation task, or any abnormality occur during the simulation, messages explaining the problem are written to the *Output List* during the process execution. These messages are prefixed either by "*Error:*", "*Warning:*" or "*Note:*" signifying something about their severity, as follows:

**Error:** A problem has occurred that makes it impossible or undesirable to continue the simulation. The simulation is aborted in a controlled manner.

**Warning:** A problem has occurred that may affect the simulation results, although the simulation itself continues. However, one should be more critical to obtained results when warnings occur, and consider changing the model.

**Note:** The event causing this kind of message is usually of no significance for the results. The message informs the user that an action has been made to perform a specific task, etc.

When a simulation process aborts with *Error*:-messages, you will also see the message *See <filename.res> for further details*. That means that sometimes (but not always) there are further error messages on the *.res* file explaining the problem. Note, however, that these additional messages are often of a low-level character and harder to understand for the average user. The main rule is that messages written to the *Output List* view should be sufficient to understand the scope of the problem.



**TIP:** You can view the *.res* file from a simulation process using the Result file browser, see [Section 10.2.1, "The Result File Browser dialog"](#).

8

A solver error will often create several messages on the *.res* file, all related to the same problem/incident. This typically happens if an exception or error occurs deep inside a program module and from there writes an error message on what is wrong (basing the message on the knowledge available to that module.) As the program trace-back unfolds, more messages on the same problem may be produced, providing more information on the problem as it becomes available. These higher-level messages can be more user friendly due to more available information. Some messages on the lowest level will have meaning for the program developers only. It is, however, often necessary to have both the higher level and the lower level messages to get full insight into the problem.

So, to understand the problem quickly, it is therefore helpful to read these messages in reverse order, starting with the last message (the highest level) and ending with the first one (lowest level). Sometimes you may ignore the lowest level messages (those output to the *.res* file only) and still see what caused the problem.

## 8.16 Using simulation events

In many analysis projects using Fedem, the task is to investigate the response on a given structural model, or a set of almost identical models, for a large set of load cases, or events. To facilitate such projects, Fedem has introduced the concept of *Simulation events*.

A simulation event is a small permutation of the structural model, where only a few model parameters, (such as a load amplitude, a spring stiffness, a cross section area, etc.) have different values. Each event is then solved and assigned a separate result database, all within the same model. That way it is efficient to continue working on the model without needing to duplicate the input data more than necessary, when setting up the different load cases. The simulation events can both be run individually, one-by-one, or all together in a large batch execution.

### 8.16.1 Setting up simulation events

When you have established the main part of the model, or at least that part that is subjected to variation in the multiple load case study, you can create the simulation events. That is done through the import of an event definition file, by selecting **Import events...** from the *Mechanism* menu.

#### Event definition file

The event definition file is an ASCII file that must be created in a text editor. It contains a tabular definition of the simulation events, where keywords representing the model quantities subjected to variation, are the same as those found in the Fedem model file (.fmm). A sample event definition file is show below.

```
: sample event definition file for Fedem.
=====
FUNC_SINUSOIDAL 1
EVENTS  FREQUENCY      AMPLITUDE      PROBABILITY EVENT_NAME
1       3.0            2.0            0.23          "my first event"
2       5.0            3.2            0.57          Second event
3       2.1            2.4            0.81          3rd event
10      4.3            0.9            0.21          some event

ANALYSIS 1
EVENTS  END_TIME
1       100.0
2       210.0
3       150.0
```

All lines starting with non-letter characters are regarded as comments.

- The first non-comment line should specify the object for which one or more model quantities are to be varied in the event. The object is specified via its model file keyword followed by the user ID.
- Then comes the event specification for this object:
  - The first line must begin with the keyword EVENTS, and then the names of the data fields to be varied for this object are listed (FREQUENCY and AMPLITUDE in the sample above).
  - Then one line for each event follows with the actual data.
- The first column becomes the user ID of the generated event objects.
- The event probability may be added in a separate column (the fourth column in the sample above) for each event. The probability is used in Fatigue analysis as a weighting factor, when calculating the expected life time of a component based on all the simulated load cases.
- The events may optionally be named (the fifth column in the sample above). The event name becomes the Description of the event, visible in the *Objects* list of the Model Manager panel.
- If more than one object is subject to variation, a new, similar table may follow with the EVENTS numbers repeated.



**TIP:** Use the Object browser (see [Section 2.4.6, "Object Browser"](#)) to see the model file keyword and associated data field names pertaining to the different objects types.

## 8.16.2 Event definition dialog

Once you have imported an event definition file, you can view the definitions of the imported events by selecting **Event Definitions...** from the *Mechanism* menu. The Event Definitions dialog box (shown below) is then opened. It contains a tabular descriptions of the simulation events, with a single row for each event and the data fields to be modified as columns. The column headings resembles the event definitions file.

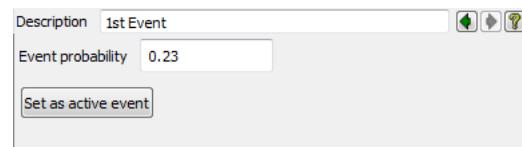
Events		FcANALYSIS 1	FcfSINUSOIDAL 1	FcfCONSTANT 1		
ID	Descr.	Prob.	END_TIME	AMPLITUDE	FREQUENCY	CONSTANT
1	my first event	0.23	120	2	3	0.4
2	Second event	0.57	140	3.2	5	0.8
3	3rd event	0.81	160	2.4	2.1	0.3
10	some event	0.21	100	0.9	4.3	
Close						

Use this dialog box to verify that your events have been correctly defined before the simulations are started, or to review the parameters for the different events without having to look in the event definition file itself.

### 8.16.3 Simulation event properties

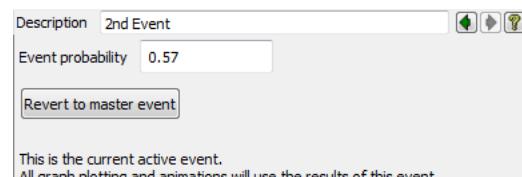
The Property Editor panel for the simulation events is show below. Only the *Description* and *Event probability* fields can be edited in this panel after the event has been created.

By pressing the **Set as active event** button, the mechanism elements are modified according to the definition of the selected event.



Subsequent solve commands and post-processing (graph plotting and animations) will then be associated with this event.

The **Revert to master event** button will restore the model to how it was without the modifications imposed by the current active event. Solving and post-processing will then be associated with the 'master event'.



### 8.16.4 Solving multiple events and result organisation



Each event can be solved and post-processed individually by using the **Set as active event** button in the Property Editor panel, as described above in [Section 8.16.3, "Simulation event properties"](#). However, it is also possible to start all events simultaneously. This is done by clicking the **Solve all Events** button, or selecting **Solve Events** from the *Solve* menu. Each event is then activated in turn, and solved, either sequentially or in parallel, depending on the *Max concurrent processes* setting in the Additional Solver Options dialog box, see [Section 8.2.3, "Additional solver options"](#).

As the different events are being solved, they get an icon in the *Objects* list of the Model Manager panel indicating that they have simulation results, as shown to the right. Here, the two first events already have got some results whereas the third event is still without results.





**NOTE:** The change of event icon is the only user feedback during simulation of multiple events, in addition to any messages in the Output List view. It is not possible to simultaneously plot results or view animations while the events are being solved. Only when event currently defined as the active event is being solved, one can simultaneously view the progress of that one.



**NOTE:** The **Solve Events** command works as the **Solve All** command (see [Section 8.2.2, "Solvers tool bar"](#)) on the individual events. Therefore, stress recovery, mode shape recovery, etc., will also be performed, if enabled, on each event. If that is not desired, you have to deactivate those analyses first by setting their start times larger than the stop time in the appropriate setup dialog boxes.

### Managing simulation event results

The structure of the results database on disk is described in [Section 10.3, "RDB directory structure"](#) in general. However, when simulation events are being used, we get one extra directory level. Directly under the `[modelname]_RDB` directory, you will have directories named `event_###` and under each of them the normal response directories `response_####`, as described in [Section 10.3.2, "Response directory structure"](#).

It is possible to postprocess the simulation results (graph plotting and animation) for the active event only. Therefore, it is not possible to, for instance, define a graph where a result quantity from two different simulation events are compared. To do that, you will have to export the curves to file first for each event, and then import them again in to the same graph (see [Section 9.2.11, "Export of Curve Data"](#) and [Section 9.2.12, "Importing Curves and Graphs"](#) for how to export and import curve data).

The commands for deleting results described in [Section 8.12, "Deleting results"](#) apply only on the current active event when simulation events are present in the model. To delete results for all events, use the **Delete Event Results** button in the **Solvers** tool bar (or **Solve** menu). This will work as if the **Delete Results** command is invoked on each individual event.





# Chapter 9 Postprocessing Results

This chapter introduces the options for postprocessing the results calculated by the various Fedem solvers. You will learn how to set up graphs, curves and animations.

Sections in this chapter address the following topics:

- [Postprocessing environment](#)
- [Graphs](#)
- [Beam diagrams](#)
- [Graph groups](#)
- [Animations](#)
- [Viewing animations](#)

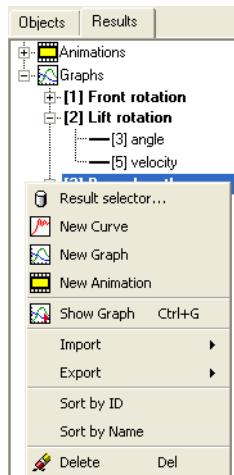
## 9.1 Postprocessing environment

In Fedem, postprocessing means evaluating the data collected from the mechanism analysis (see [Chapter 8, "Mechanism Analysis"](#)). The options for postprocessing are graphing and animating. You can perform these tasks using the *Results* list of the Model Manager panel and the commands available in the *Result* menu. Results are displayed in the Workspace area.

### Model Manager Results list

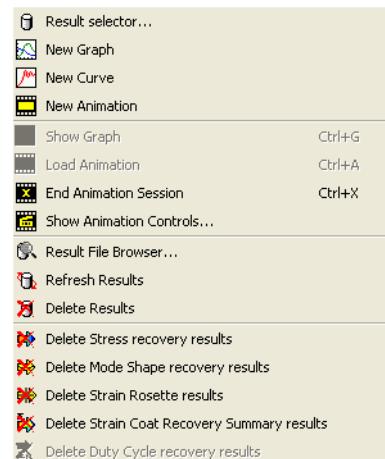
The *Results* list of the Model Manager panel (shown at right) displays the list of user-defined result views in the model. In addition to the commands available in the *Result menu* (see below), many shortcut commands can be used to manage results in the *Results* list. The shortcut menu, which is accessed by right-clicking in the Model Manager panel, displays commands relevant for the selected object only.

Each curve in the Model Manager *Result* list has an icon next to the curve name, representing its legend. The curve color and symbol can be changed in the *Appearance* tab (see [Section 9.2.7, "Appearance"](#)).



### Result menu

The *Result* menu (shown at right) contains commands for creating result views, and managing the result files in your model. Use of these commands is explained in the following sections.



### Workspace area

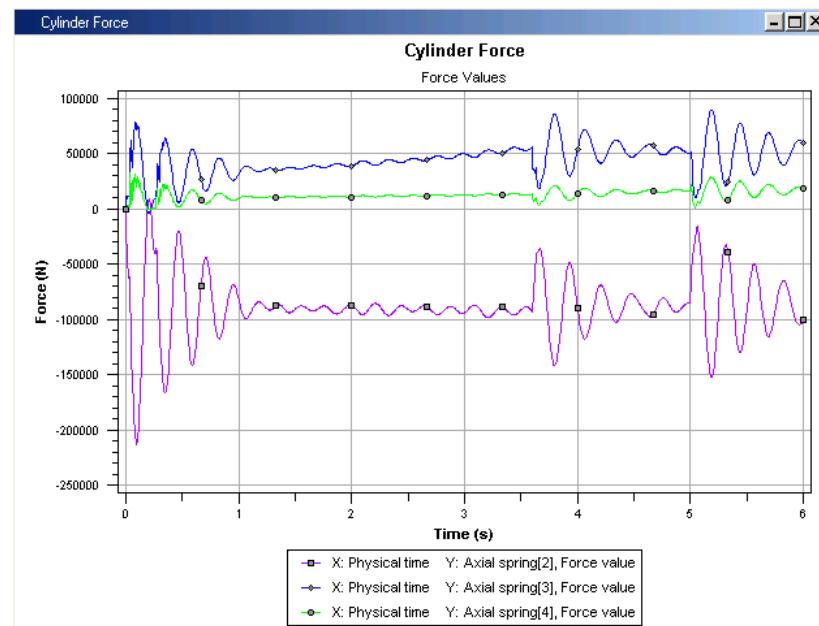
The Workspace area is used to display the graphs and animations you create.

- Graphs can be displayed in individual windows that are labeled with the user-specified description of the graph; these windows are called *Graph* views. To specify a description and display a *Graph* view, see [Section 9.2.3, "Graph properties"](#) and [Section 9.2.2, "Showing a graph"](#).
- Animations can be displayed in the *Modeler* view (one at a time). To display an animation, see "[Loading animations](#)" in [Section 9.5.1](#).

## 9.2 Graphs

To track the progress of any variable during the simulation, you can create two-dimensional graphs of the values. Each graph can contain several curves, enabling comparison of the simulation variables.

You can customize your graphs with titles, axis and data labels, and legends (as shown below).



Graphs can be set up before or after performing the dynamics simulation. If you create a graph before performing the simulation, you can observe the values during the simulation, as they are constantly updated (see [Section 8.11, "Interaction during processing"](#)). Otherwise, you can view the entire set of graphed values after the simulation is completed.

Graphs plotting values vs. time will have a time indicator bar present when a time history animation is loaded. This bar will show the current animation time in the graph. This makes it easy to see the correlation between the motion and the graphed values.

In addition to graphs plotting the evolution of result quantities during the simulation, another type of graphs exists which plots the variation of a result quantity along a beam structure at a certain time. These types of graphs are described in [Section 9.3, "Beam diagrams"](#).

### 9.2.1 Creating graphs and curves

You can create as many graphs as you like, either before or after solving the dynamics simulation. It is recommended that you provide descriptive names for your graphs, as the description is used in the Model Manager *Results* list to distinguish between graphs. The description is also used as the title of the *Graph* view when the graph is displayed in the Workspace area. To specify a description, see [Section 9.2.3, "Graph properties"](#).

Once you have created a graph, you will need to create curves (plotted data) for the graph and specify descriptions and properties for the curves.

There are two ways to create these items, either by using the **New Graph** and **New Curve** commands, or by dragging result variables from the RDB selector dialog box into the *Result* list of the Model Manager panel, as described below.

#### Graph



To create a graph, select **New Graph** from the *Result* menu, or right-click in the Model Manager *Results* list and select **New Graph**. An object titled "New Graph" is listed in the *Results* list of the Model Manager panel.

#### Curves

To create a curve, complete the following steps:



1. In the Model Manager *Results* list, select the graph to which you want to add the curve.
2. Select **New Curve** on the *Result* menu or right-click the graph in the Model Manager *Results* list, select **New Curve**.
3. Alternatively: Right-click an empty spot in the *Results* list of the Model Manager panel and select **New Curve**. This will create a new graph with a single curve.

An object named "New Curve" is then listed under the current graph in the *Results* list of the Model Manager panel.

#### Creating curves and graphs by drag and drop

When you need to create several curves and graphs containing results from the results database, or want to inspect several result values without the need to create persisting graphs for all of them, it is convenient to use the drag and drop method to create and sort curves and graphs.

To create curves or graphs by drag and drop, right-click in the *Result* list of the Model Manager panel and select **Result selector...** The RDB selector dialog box will then pop up (shown to the right).

Select the result you want to plot, and drag it from the RDB selector dialog box and onto the *Result* list of the Model Manager panel. If you drop it on an existing graph, it will be created as a curve in that graph, otherwise a new graph will be created with the selected results as a new curve.

If the graph receiving the new curve is visible, the new data is automatically loaded and displayed.

For more detailed information about the RDB selector dialog box, see "[Selecting RDB results](#)" in [Section 9.2.4](#).

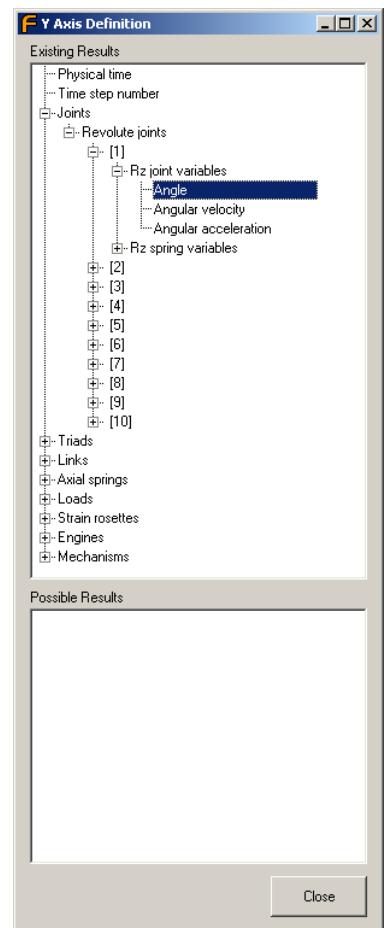
#### **Repeat a curve definition for all objects**

In some cases it is useful to look at some particular result from all the objects of a certain type in the model. E.g., to find the triad with the highest forces in the model, or to look at the acceleration levels all over the model.

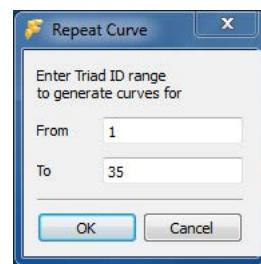


This can be achieved in a convenient way using the command labeled **Repeat curve for all objects**. This command is available in the right-click menu of the *Result* list of the Model Manager panel when right-clicking a curve. This command repeats the curve definition of the curve selected, and uses it on all the objects of the same type in the model. It also automatically assigns a color to the curves depending on the ID number of the object. The colors assigned will be black-blue-cyan, where low ID numbers makes the curve get a blackish color, while higher numbers will turn the curve blue or cyan.

Normally, it is only the object used for the y-axis value in the curve definition that is cycled, but if the curve being repeated uses the same object both for the x-, and y-axis they are cycled together.



It is also possible to repeat a curve definition for not every object of the same type in the model, but only a subset of the similar objects. This is done by using the command labeled **Repeat curve for object range...** in the right-click menu of the *Results* list of the Model Manager panel. This opens a dialog box (shown to the right) where you can enter the user ID range of the objects for which you want the curve definition to be repeated.



**NOTE:** Since the user ID numbers are unique only within each sub-assembly (see Section 5.15, "Sub-assemblies"), the **Repeat curve for object range...** command applies only to the objects within the same sub-assembly as the object that is plotted by the selected curve. On the other hand, the **Repeat curve for all objects** command always applies to all objects in all sub-assemblies (if any) of the model.

### Moving curves to a new graph

To move curves from one graph to another, simply select the curves and drag them to the new graph. This is done by pressing and holding the left mouse button while the mouse cursor is above the curves to move, then move the mouse to the target graph and release the mouse button.

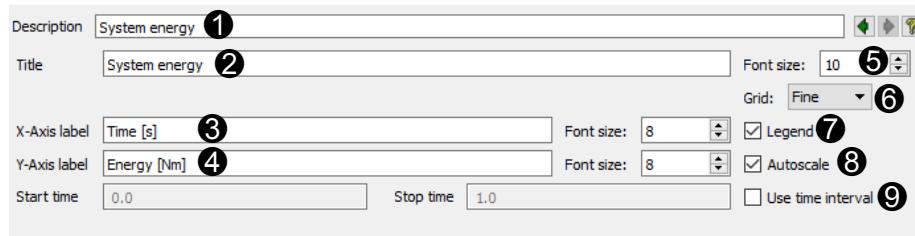
#### 9.2.2 Showing a graph



To display graphs, select one or more graphs or curves in the Model Manager *Results* list, right-click and select **Show Graph** on the shortcut menu (or select **Show Graph** on the *Result* menu). The selected *Graph* views opens in the Workspace area displaying the graphs.

#### 9.2.3 Graph properties

Now that you can create a graph and its curves, you need to specify properties for the graph. To display the properties in the Property Editor panel, select the graph in the Model Manager *Results* list. The properties for the graph are displayed in the Property Editor panel as shown below.



- 1 *Description* – An optional user-specified description that is displayed as both the graph name in the *Results* list, and the title of the associated *Graph* view in the Workspace area. It is recommended that you use a descriptive name or phrase to distinguish between similar graphs. This name is not included in the *Graph* view itself.
- 2 *Title* – You can specify a title for the graph that is displayed in the *Graph* view.
- 3 *X-Axis Label* – You can provide a label for the abscissa that is displayed in the *Graph* view.
- 4 *Y-Axis Label* – You can provide a label for the ordinate that is displayed in the *Graph* view.
- 5 *Font size* – You can specify the font size to be used on the legends and the axes ticks marks in the *Graph* view.
- 6 *Grid* – You can specify the type of grid (coarse, fine, or no grid) for the *Graph* view by selecting from this pull-down menu.
- 7 *Legend* – You can add a legend to the graph by enabling the **Legend** option. The legend names you specify for each curve are used in the legend (see [Section 9.2.4, "Curve properties"](#) below).
- 8 *Autoscale* – Enabling this option automatically scales the x- and y-axes to accommodate new curves as they are added to the graph (or when opening the *Graph* view). Disabling **Autoscale** allows you to fully control the x- and y-axis range.
- 9 *Start time, Stop time, Use time interval* – You can enable the **Use time interval** option to specify a time interval for loading the graph. Only the RDB data that fall within the specified time interval are loaded when the graph is opened or exported. This only affects curves that are plotting simulation results. Imported curves, or curves plotting an internal function are not affected by these settings.



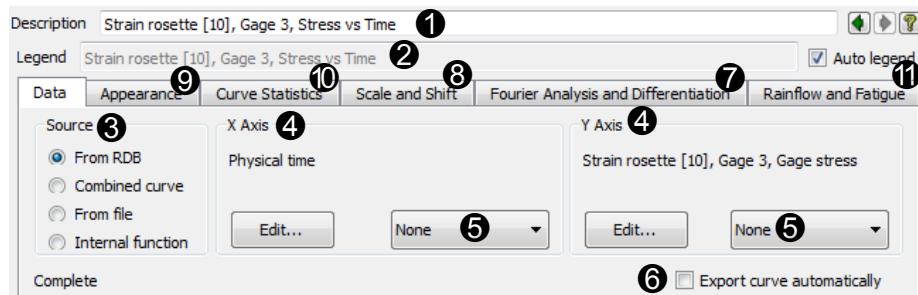
**NOTE:** When you make changes to graphs or curves in the Property Editor panel, the changes are updated dynamically in the graph view.

#### 9.2.4 Curve properties

Having completed the specification of a graph, you want to set the properties of its associated curves. To display the curve properties in the Property Editor panel, select the curve in the Model Manager *Results* list.

The curve information is organized under six tabs, as shown below: *Data*, *Appearance*, *Curve Statistics*, *Scale and Shift*, *Fourier Analysis and Differentiation*, and *Rainflow and Fatigue*.

The *Data* tab is used to specify the data to be plotted and is described below. The other five tabs are discussed in the subsequent sections.



- 1** *Description* – this is used as the name for the curve in the Model Manager *Results* list. The description is updated automatically when changing the x- or y-axis definitions, unless it has been edited manually in the mean time. This name is not used in the *Graph* view.



**TIP:** If you have manually edited the description field of a curve, you can at any time return to the auto-generated description by deleting it completely.

- 2** *Legend* – if you choose not to supply a name for the curve, you can enable **Auto legend** to use a combination of the x- and y-axis labels for the legend. When **Auto legend** is used, the legend will always be equal to the auto-generated description of the curve.



**NOTE:** The **Legend** option must be enabled in the *Graph Properties* panel to display the legend in the *Graph* view (see [Section 9.2.3, "Graph properties"](#)).

- 3** *Source* - the data plotted in a curve originate either from the results database (*From RDB*), from an external file (*From file*), or from a Fedem function (*Internal function*). In addition, the data can be defined via a mathematical expression with some of the other curves as variables (*Combined curve*). The Property Editor panel shown above is that of a "From RDB" curve. The panels of the other curve types are discussed below (see "[Creating curves from file](#)", "[Creating curves from a function](#)", and "[Creating combined curves](#)" below).



- 4** *X Axis / Y Axis* – to define the values used for the x- and y-axes in a "From RDB" curve, click the **Edit...** button. The RDB selector dialog box is then opened allowing you to pick the wanted results (see "[Selecting RDB results](#)" below for details). Alternatively, one may also right-click a curve in the Model Manager *Results* list and then select **Edit X Axis...** or **Edit Y Axis...**

**TIP:** While editing the x-axis, you can click **Apply** in the RDB selector dialog box, then click the **Edit...** button for the y-axis, and start selecting for the y-axis.

- 5** *Result Operation* – the pull-down menus next to the **Edit...** buttons list mathematical operations (such as extracting the x-component or computing the length of a vector) related to the result quantities selected for the x- and y-axis. If the result quantity selected in Step 4 above is a *Position matrix* (of either a Triad or Part), the menu will also contain several angular quantities which can be derived from the position matrix. See "[Derived angular quantities from position matrices](#)" below for further details on these items.



**NOTE:** If you make a change to the Result Operation, the curve changes dynamically in the graph view.

- 6** *Export curve automatically* – toggles whether the dynamics solver shall export this curve automatically, to the file specified in the Dynamics Solver Setup dialog box. See "[Output tab](#)" in [Section 8.5.2](#).
- 7** *Fourier Analysis and Differentiation* - this tab allows you to perform a Fast Fourier Transformation of the curve data. See [Section 9.2.5, "Fourier analysis, differentiation and integration"](#). You may also plot the derivative or the integral of the curve data via this tab.
- 8** *Scale and Shift* - this tab allows you to apply scaling and shift to the curve data. See [Section 9.2.6, "Scale and Shift"](#).
- 9** *Appearance* - this tab allows you to change appearance of individual curves. See [Section 9.2.7, "Appearance"](#).
- 10** *Curve Statistics* - this tab allows you to extract statistical properties for individual curves. See [Section 9.2.8, "Curve Statistics"](#).
- 11** *Rainflow and Fatigue* - this tab allows you to perform a Rainflow calculation and to assess fatigue results based on the curve data. See [Section 9.2.9, "Fatigue calculation from standard S-N curves"](#).



**NOTE:** The curve is not displayed in the graph view until you have fully defined the curve variables (indicated by the word "Complete" appearing at the bottom of the Property Editor panel). The word "Incomplete" appears at the same location until the curve is properly defined.



**TIP:** Once you have created and fully defined curves, you can select them directly in the Graph view, or clicking a curve in the Results list of the Model Manager panel. When selected, curves are highlighted in red.



**TIP:** You can drag a curve from one graph and drop into another in the Model Manager Results list.

## Selecting RDB results

To select the results to plot as x- and y-axis valued for a curve, complete the following steps:

1. In the Model Manager *Results* list, select the curve you want to edit. Its properties are displayed in the Property Editor panel.
2. In the Property Editor panel, click the **Edit...** button for the X-Axis, or right-click on a curve in the Model Manager *Results* list and select **Edit X Axis...** or **Edit Y Axis...** The RDB selector dialog box (shown at right) is then opened.
3. Select a mechanism element from the *Existing Results* list or in the *Modeler* view.



**NOTE:** If the mechanism analysis have been performed, variables for the selected element are listed in the Existing Results list. If you have not yet performed an analysis, variables are listed in the Possible Results list.

4. Select a variable to be used as the x-axis quantity from either the *Existing Results* or *Possible Results* lists, and click **OK** to close the panel or **Apply** to continue variable selection.



**CAUTION:** Some of the variables listed in the Possible Results list may not be present in the results database. For example, a joint may or may not have a spring or damper attached at each DOF, but in the Possible Results list all possible springs and dampers are listed - one for each joint DOF. If such a nonexistent variable is selected, the associated curve does not appear in the graph view.

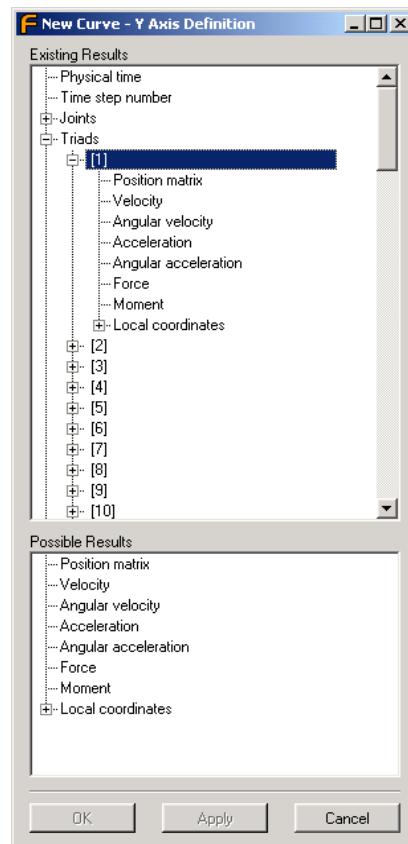


**NOTE:** Variables such as Physical Time and Time Step Number are not associated with a mechanism element, and are listed independently in both lists.



**NOTE:** Categories such as Revolute Joints or Z-Rotation Joint Variables cannot be selected as variables; only those items in the expanded lists (such as Angular Deflection) can be selected for use as variables.

5. Repeat steps 2 through 4 to select the Y-Axis variable.





**TIP:** To easily find out what result quantities, if any, that already have been plotted for a given mechanism object, just select the object and inspect the Topology view (see [Section 2.4.4, "ID and Topology panel"](#)). The curves plotting quantities in the selected object are then listed under the Plotted by: heading.

### Derived angular quantities from position matrices

Totally 10 derived quantities may be plotted for a *Position matrix* result item (as shown to the right). The *Euler Angle* quantities (top three) are the indicated angles computed from an imagined incremental rotation from the global coordinate system axes to the orientation represented by the position matrix. The bottom three items are similar quantities computed from a Rodriguez parameterization of the incremental rotation. See the Fedem 7.6 Theory Guide, *Section 2.3, "Finite Rotation"* for the definition of these angular quantities.

Euler Angle ZYX X
Euler Angle ZYX Y
Euler Angle ZYX Z
Position Length
Position X
Position Y
Position Z
<b>Rotation Angle X</b>
Rotation Angle Y
Rotation Angle Z

### Plotting internal control variables

When you plot internal control variables, you will probably discover that some of the control lines don't have any results. The reason for this is that more than one control line share the same control variable, and the results appear on only one of these (usually the one with the lowest ID). This situation will occur when one element's output is used as input to more than one other element.



**TIP:** Have the Control Editor view open during curve result selection. If you select a control line in the RDB selector dialog box, that line will also be highlighted in the Control Editor view. Vice versa, if you select a control line in the Control Editor view, that control line will be selected in the RDB selector dialog box. That way you can easily see which control line you will have to plot to get the variable you want.

9

### Creating curves from file

A curve can be created from an external file by selecting the *From file* option on the Property Editor panel's *Data* tab. The panel used to define such a curve is shown below.

Data	Appearance	Curve Statistics	Scale and Shift	Fourier Analysis and Differentiation	Rainflow and Fatigue
Source <input type="radio"/> From RDB <input type="radio"/> Combined curve <input checked="" type="radio"/> From file <input type="radio"/> Internal function	File <b>1</b> <input type="button" value="Browse..."/> <input type="button" value="Reload..."/> <b>3</b> Column/channel <b>4</b> <input type="button" value="Not set"/> <input type="button" value="Select..."/> <b>5</b>	<input type="checkbox"/> Incomplete <input type="checkbox"/> Export curve automatically			

- ① **File** field - The selected curve data file will be shown here.
- ② **Browse...** button - Opens a dialog box for selection of curve data file. The supported file formats are ASCII, nCode DAC and MTS RPC III.
- ③ **Reload** button - If your data file has changed, you can click this button to reload the curve into the viewer.
- ④ **Column/channel** field - The name of the selected channel will appear here. (Only applicable for multi-column ASCII and MTS RPCIII files.)
- ⑤ **Select...** button - If you imported a multi-column ASCII file, or a MTS RPCIII file, you have to select which column or channel to extract data from. A dialog box for doing so will appear when clicking this button.

ASCII curve data files may consist of two or more columns of data. The columns can be separated by either white-space (one or more tab- or space-characters) or the comma-character (,). The first column is taken as the x-axis values. If more than two columns are present in the file, the actual column to use for the y-axis values is selected from the dialog box appearing when clicking the **Select...** button. The columns are here identified by numbers (1 up to number of columns minus 1, the first column of the file is not listed). However, it is also possible to include user-defined descriptions of the columns. This is done by inserting as the first line of the file, the string #DESCRIPTION followed by a tab-separated list of headings for each column (not for the first column).



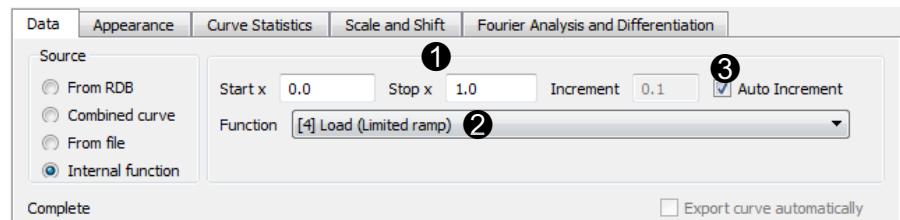
**CAUTION:** It is not possible to use comma (,) as decimal point in ASCII curve data files, as this character is interpreted as a column separator. Only (.) is valid. Therefore, if you have data files exported from, e.g., a Norwegian version of Microsoft Office Excel, make sure that the decimal point is correct before importing the file into Fedem.

The nCode DAC and MTS RPC III formats are proprietary binary formats. DAC support single-columns files only whereas RPC supports multiple columns (or channels). Please refer to documentation from nCode and MTS for further details on these formats.

And alternative way of creating curves from file, is to import multiple curves into an existing or a new graph, see [Section 9.2.12, "Importing Curves and Graphs"](#).

### Creating curves from a function

A curve can be created from one of the functions defined in your model by selecting the **Internal function** option on the Property Editor panel's *Data* tab. The panel used to define such a curve is shown below.



- ① *Start x, Stop x, Increment* - Sets the end points for the domain to plot, and the rate at which the function is sampled.
- ② *Function* - The function to be plotted is chosen from this pull-down menu. The menu lists all functions currently in your Fedem model.
- ③ *Auto Increment*- Most functions have the option to have the resolution set automatically. If used, Fedem will determine which points are sufficient to describe the function and plot those. In that case the specified increment is not used.

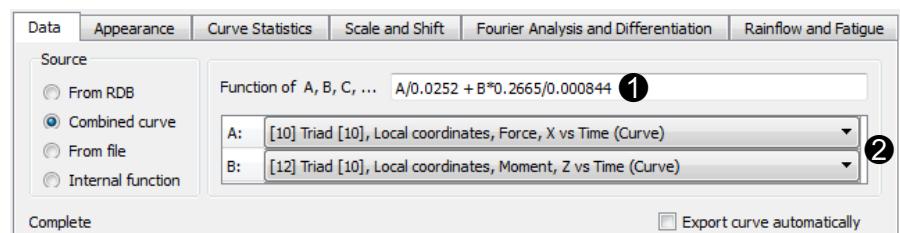


**NOTE:** If plotting a Poly-line function with **Auto Increment** set, the Start x and Stop x field values are not used either. In this case, the x-axis domain is automatically adjusted to fit the curve point data.

9

### Creating combined curves

A curve can be created as a combination of any of the (up to 10) other curves currently defined in your model by selecting the *Combined curve* option on the Property Editor panel's *Data* tab. The panel used to define such a curve is shown below.



- ① *Function of A, B, C, ...* - In this field you may type in a mathematical expression using the upper case letters in range [A,J] as variables representing other curves in the model. Up to 10 other curves may be

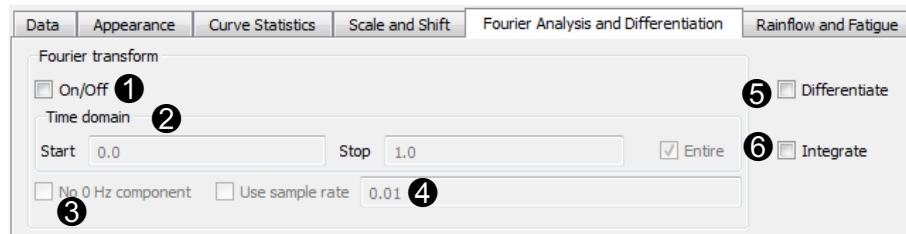
used in combined curve definition. The syntax of the mathematical expression follows that of the Math expression function types, see "[Math Expression](#)" in [Section 5.11.5](#).

- ② Depending on how many of the variables A-J you have specified in the expression field, a number of pull-down menus labeled A:, B:, etc., are provided here, in which you can assign a curve to each variable. Curves of any type may be selected here, also other *Combined curves*, except for this curve itself or other combined curves referring to this curve either directly or via other combined curves.

The *Combined curve* feature is an efficient way of plotting result quantities that are not stored directly in the results database. For instance, if you want to plot the stress at a certain point in the beam element you know that this is just a linear combination of the two bending moments and the axial force at that point along the beam. You can then set up this linear combination as an expression where the coefficients involved depends on the cross section geometry, and using the variables A, B and C to represent the sectional forces involved. Then you define three other curves plotting these quantities and select them in the corresponding pull-down menu in the *Combine curve* properties.

### 9.2.5 Fourier analysis, differentiation and integration

Options for performing a Fast Fourier Transform (FFT) of the curve data are found on the *Fourier Analysis and Differentiation* tab of the Property Editor panel. The *Fourier Analysis* properties are shown below.



- 1** *Fourier transform On/Off* - When toggled *On* the plotted curve is replaced by its discrete Fourier transform (the Fourier transform is a representation of the curve in the frequency domain). The value plotted is the magnitude of the transform.



**NOTE:** The scale and shift parameters specified for the curve (see [Section 9.2.6](#), "[Scale and Shift](#)") are applied to the curve data **before** the transform is computed. You may **not** scale or shift the transformed curve.

- ② **Time Domain** - What part of the curve to transform is specified using the Time Domain options. If **Entire** is toggled on, then data for the curve's entire domain is used. If **Entire** is toggled off, then a start and stop time may be set in the fields labeled *Start* and *Stop*.
- ③ **No 0 Hz component** - The arithmetic mean of the original curve data is reflected in the transform's value at 0 Hz (the transform's first point). To facilitate transform analysis you might want to "cancel out" the 0 Hz component by using this option.
- ④ **Use sample rate** - The sample rate used in the transform is by default equal to that of the curve data. However, if the curve has a non-constant sample rate the transform will fail. In such cases the wanted sample rate may be input using this field.
- ⑤ **Differentiate** - When this toggle is enabled, the plotted curve is replaced by its derivative, which is computed from the curve point values  $f_i = f(x_i)$  through the formula

$$f'(x_i) = \frac{1}{2} \left( \frac{f_{i+1} - f_i}{x_{i+1} - x_i} + \frac{f_i - f_{i-1}}{x_i - x_{i-1}} \right)$$

- ⑥ **Integrate** - When this toggle is enabled, the plotted curve is replaced by the integral of the curve data, which is computed from the curve point values through the recursive formula

$$\int_{x_0}^{x_i} f(x) dx = \int_{x_0}^{x_{i-1}} f(x) dx + \frac{1}{2}(f_i + f_{i-1})(x_i - x_{i-1})$$



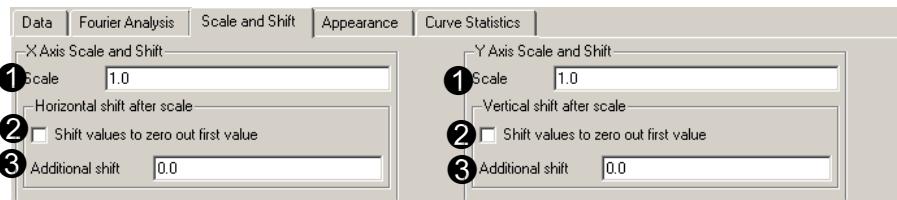
**CAUTION:** The Fourier transform needs to be recalculated each time points are added to the curve. Consequently, if curves are plotted while the Dynamics solver is run, then transforming these curves will increase the CPU load during solving.



**TIP:** A good reference on the theory of Fourier transforms is: W. Rudin, "Real and Complex analysis", McGraw-Hill, 1974.

## 9.2.6 Scale and Shift

Options to scale and shift the curve data are found on the *Scale and Shift* tab of the Property Editor panel. You may apply scaling and a shift on the curve data independently in the x- and y-directions. The *Scale and Shift* properties are shown below.



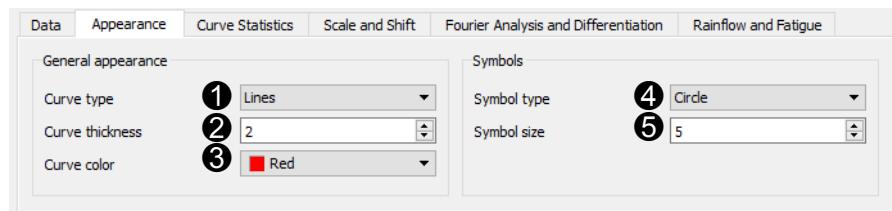
- ① **Scale** - Scale factor applied to the x- or y-axis values.
- ② **Shift values to zero out first value** - For the x-axis; shift the curve horizontally such that its first x-value becomes zero. For the y-axis; shift the curve vertically such that its first y-value becomes zero.
- ③ **Additional shift** - Additional horizontal/vertical shift (i.e., in addition to the zero-out operation, if applied) in the curve's x/y-values.



**NOTE:** Scaling the y-axis of a curve will affect the results of fatigue damage calculations (see [Section 9.2.9, "Fatigue calculation from standard S-N curves"](#)).

## 9.2.7 Appearance

The Curve appearance can be altered by selecting the *Appearance* tab in the Property Editor panel. The associated properties are shown below.

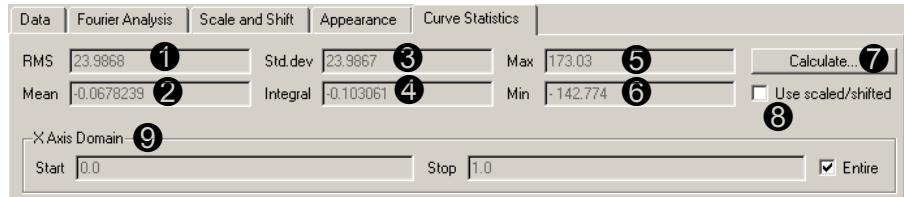


- ① **Curve Type** – You can select *Lines*, *Dots*, or *Invisible* from the *Curve Type* pull-down menu.
- ② **Curve thickness** – You can adjust the curve thickness with 5 different levels, from 0 (thinnest) to 4 (thickest).
- ③ **Curve Color** – Use the drop-down menu to select a different curve color. You may either select one of the pre-defined colors, or create a new by selecting **More...** at the bottom of the drop-down menu.

- ④ **Symbol type** – You can select a symbol (cross, circle, triangle, etc.) to display on the curve. A symbol will be shown on all points of the curve, so use this with care if it consists of a huge number of points.
- ⑤ **Symbol Size** – Use the spin box to control the size of the symbols.

### 9.2.8 Curve Statistics

On the *Curve Statistics* tab you can display different statistical properties of a curve. The associated properties are shown below.



- ① **RMS** - The Root Mean Square value, found from

$$y_{rms} = \sqrt{\frac{\sum_{i=1}^n y_i^2}{n}}$$

- ② **Mean** - The Mean value,  $\bar{y}$ .

- ③ **Std.dev.** - The Standard Deviation (biased), found from:

$$\sigma = \sqrt{\frac{\sum_{i=1}^n (y_i - \bar{y})^2}{n}} \quad \text{where } \bar{y} \text{ is the mean}$$

- ④ **Integral** - By the Trapezoid rule.

- ⑤ **Max** - The overall maximum of the y-values in the curve data set.

- ⑥ **Min** - The overall minimum of the y-values in the curve data set.

- ⑦ **Calculate...** - Press this button to retrieve the statistical values.

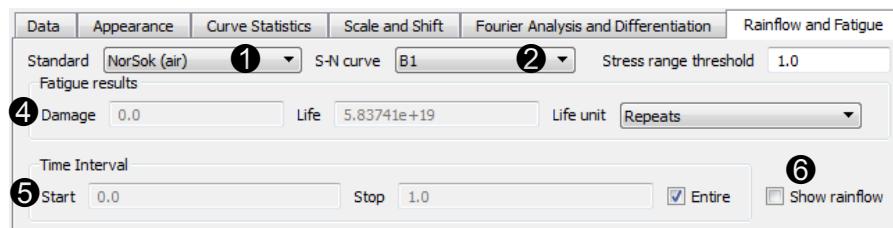
- ⑧ **Use scaled/shifted** - Toggling on this button will make the calculation take into account any scale and shift values defined in the *Scale and Shift* tab (see [Section 9.2.6, "Scale and Shift"](#)), and in effect do the calculations on the curve as it is shown in the graph view. If the button is toggled off, the unprocessed data will be used.

- ⑨ **X Axis Domain** - Toggle the **Entire** button on to use all the data points on the curve, or specify a start and a stop value. If you specify an interval, two vertical lines at the start and stop values, will appear in the graph view when you click the **Calculate...** button.

9

### 9.2.9 Fatigue calculation from standard S-N curves

Options to assess fatigue results based on plotted stress histories are found in the *Rainflow and Fatigue* tab of the Property Editor panel. The associated properties are shown below. Here, you may select different standard S-N curves to base the damage calculation on, specify a time interval for the damage calculation, and evaluate the equivalent life in days, hours or repeats. For details on how the damage is calculated from a given time history response, see the Fedem 7.6 Theory Guide.



- ① **Standard** - Select the fatigue standard to use in the calculation.
- ② **S-N curve** - Select an S-N curve from the selected standard.
- ③ **Stress range threshold** - Stress ranges with magnitude below this threshold are ignored in the stress cycle counting (rainflow analysis).
- ④ **Damage, Life, Life unit** - The Results frame displays the damage results for the plotted stress history. The life is displayed in days, hours or repeats, depending what you select from the *Life unit* drop-down.
- ⑤ **Start, Stop, Entire** - You can specify what part of the curve to use for fatigue calculations by setting the *Time Interval* options. If *Entire* is toggled on, data for the entire domain of the curve is used. If *Entire* is toggled off, the time interval is specified by the *Start* and *Stop* fields.
- ⑥ **Show rainflow** - When this toggle is enabled, the plotted curve is replaced by the results of the Rainflow calculation, i.e., it shows the magnitudes of the stress ranges that are the basis of the subsequent damage calculation.



**NOTE:** If the y-axis of the curve is scaled, the scaling factor will be applied to the damage calculations too (see [Section 9.2.6, "Scale and Shift"](#)).

### 9.2.10 View control

In graph views, you can manipulate the display using the *Zoom and Pan* tool bar shown below. For the use of these commands, see [Section 2.6.4, "Zoom and Pan"](#).



You can also use the Dynamic Pan  $\leftarrow$  (**F1**) and Dynamic Zoom  $\leftarrow\uparrow$  (**F2**) commands, in the same way as they are used in the *Modeler* view (see [Section 2.6.1, "3D Navigation"](#)).



**TIP:** The following tips are useful for viewing graphs:

- If the number of points in your graph is very high, dynamic panning and zooming can take a long time; use the commands on the *Zoom and Pan* tool bar instead to speed up graphic performance.
- You can use the **Zoom Window** command to zoom in on a small area, enabling you to see the curves in that area with greater detail. This command is turned on whenever the **Z** key is pressed while viewing a graph.
- You can use the **Zoom Window With Autoscale** command to zoom in on an area. The contents inside the zoom rectangle will be scaled to fit inside the graph view. This command is turned on whenever the **X** key is pressed while viewing a graph.
- To adjust the graph axes so that the entire curves fit into the graph view, click the **Zoom All** button, or press **F5**.

### 9.2.11 Export of Curve Data

Curve and Graph objects in Fedem can be exported to files for further processing in external software. We distinguish between exporting graphs and exporting curves.

#### Curve export

When exporting one or more curves, each curve is written to a separate file. The file format can be either Single Column ASCII, nCode DAC or MTS RPCIII time history file.

To export curves, select the curve or curves you want to export in the Model Manager *Results* list, right-click and select **Export** -> **Export Curves**... A dialog box will then pop up. If you have selected only one curve, you can select location and file name of the exported curve. If you have selected several curves, you must select a directory to export to. If you select a graph, all its curves will be exported.

When you select a directory to export to, the files will be given names automatically. The file name will be on the form:

*G\_<OwnerGraphID>\_C\_<CurveID>\_<CurveDescription>.<Format>*



**TIP:** Curves that plot result data from the Dynamics Solver may also be exported automatically when the solver has finished (see "[Output tab](#)" in [Section 8.5.2](#)).



**NOTE:** The exported data is equal to the results from the settings in the Fourier Analysis tab and the Scale and Shift tab. If you want to export unprocessed data, go to these tabs and set all the values back to default (see [Section 9.2.5, "Fourier analysis, differentiation and integration"](#) and [Section 9.2.6, "Scale and Shift"](#)).

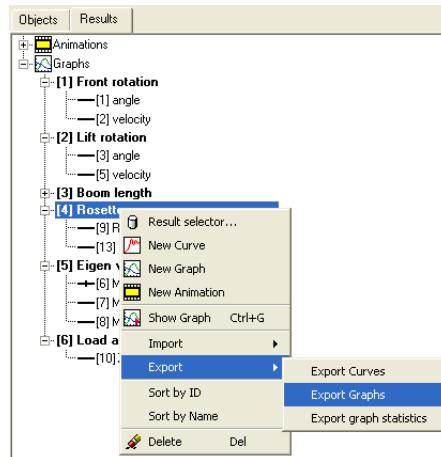
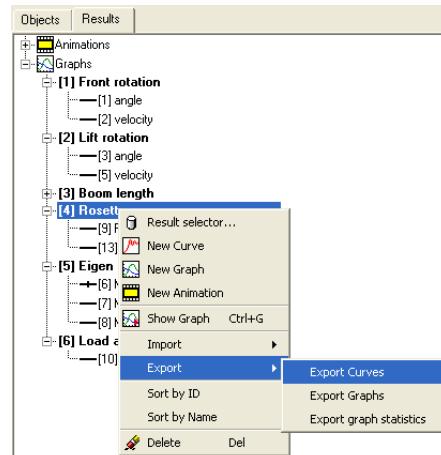


**CAUTION:** When exporting to nCode DAC or MTS RPCIII,  $\Delta x$  needs to be constant across the entire data set. Using either Physical Time with constant time step size, or Time Step Number will satisfy this requirement.

### Graph export

When exporting one or more graphs, each file exported will contain several curves. The file format is either Multi Column ASCII or MTS RPCIII time history file.

Select the graph or graphs you want to export in the Model Manager *Results* list, right-click and select **Export** -> **Export Graphs**... A dialog box will then pop up, with slightly different appearance depending on what you have selected.



If you have selected one graph, or a collection of curves belonging to the same graph, the dialog box will let you select directory, file name and format for the exported graph file. If you selected only a subset of curves from a graph, only the selected curves will be written to the graph file.

If you have selected several graphs and/or curves belonging to different graphs, the dialog box will let you specify a directory to write the files to and the file format. One file is written to the specified directory for each selected graph. The name of each file is assigned automatically, and will be on the form

`G_<GraphID>_<GraphDescription>.<Format>`



**CAUTION:** When exporting graphs, all curves in the selection must have equal x-axis definitions, both in terms of number of data points and increment  $\Delta x$ . When exporting to Multi-Column ASCII, the x-axis values of the exported graph are thus set identical to those of the first curve in the selection. Subsequent curves having a lower resolution in their data sets will then be interpolated, where needed; if they have a higher resolution, some data points will be omitted in the exported graph. When exporting to MTS RPC III, the constant increment  $\Delta x$  for the graph is selected as the smallest increment between two data points among all the selected curves. Curves having a lower resolution than this increment will then be interpolated, where needed.



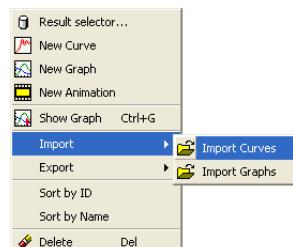
**TIP:** To export a single curve or graph, you may also use the **Export** and **Export Object...** item in the File menu, after selecting the desired curve or graph.

## 9.2.12 Importing Curves and Graphs

Curve data can be imported into Fedem both as single curves, and complete graphs.

### Importing Curves

To start importing curves, right-click with your mouse on a graph, a curve or on an empty spot in the Model Manager *Results* list and select **Import** and **Import Curves**. In the dialog box that pops up, select one or more files you want to import. One curve is created for each file. For details on the supported file formats, see "[Creating curves from file](#)" in [Section 9.2.4](#).



If you right-clicked on a graph, the curve data will be imported into that graph, whereas if you clicked on a curve, the data will be imported into the graph containing that curve. If you clicked on an empty spot, a new graph will be created for the new curve(s).

### Importing Graphs

To start importing graphs, right-click with your mouse anywhere in the Model Manager *Results List* and select **Import** and **Import Graphs**. In the dialog box that pops up, select one or more graph files you want to import. One graph is created for each of the selected files, containing one curve for every column or channel in the file.

#### 9.2.13 Exporting to picture files

You may also want to export a graph view to a picture file. This is accomplished by first selecting the *Graph* view you want to export and then selecting **Export** and **Export View...** from the *File* menu. You may choose to output in either BMP, JPEG or PNG file format. See [Section 2.9.1, "Exporting a part"](#), for more about the export capabilities in Fedem.

#### 9.2.14 Printing graphs

When a *Graph* view is active, you can send its contents directly to a printer for printing. Select the printer symbol on the tool bar, or the **Print With Setup** command in the *File* menu. When selecting the latter, you will be able to select which printer to use, paper format and so on.

## 9.3 Beam diagrams

In addition to viewing the evolution of a result quantity at a fixed point in the model as a function of time (or any other fixed simulation quantity), as described above, it is often desirable to view the variation of a quantity along a one-dimensional structure, such as a beamstring, at a given time. For this purpose, we offer the possibility to create *Beam diagram* graphs.

#### 9.3.1 Creating beam diagrams



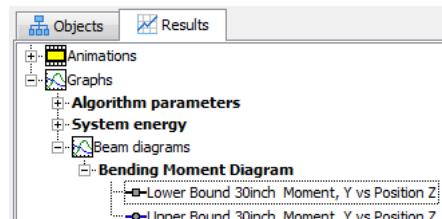
The only way to create a beam diagram graph is to select **New Beam diagram** from the *Result* menu, or right-click in the Model Manager *Results* list and select **New Beam diagram**. There is no drag-and-drop creation option here, as for the regular graphs. The created graph titled "New Graph" is then listed in the *Results* list of the Model Manager panel.



To create curves in a beam diagram graph, select the graph to which you want to add the curve and select **New Curve** from the *Result* menu or the right-click menu in the Model Manager *Results* list. The created curve titled "New Curve" is then listed under the selected beam diagram graph.

The description and other properties of the beam diagrams and their curves may be edited in the same way as for other graphs and curves.

All beam diagram graphs are grouped under the tree-node labeled “Beam diagrams” in the Results list (shown to the right). This is to make the distinction of these type of curves compared to all the other curve types more clear. It is not possible (and makes no sense either) to mix these types of curves with the other curves within the same graph. A curve in a beam diagram can only be copied or moved into an other beam diagram graph, and not into the other graphs. Similarly, a regular curve can not be copied or moved into a beam diagram graph.

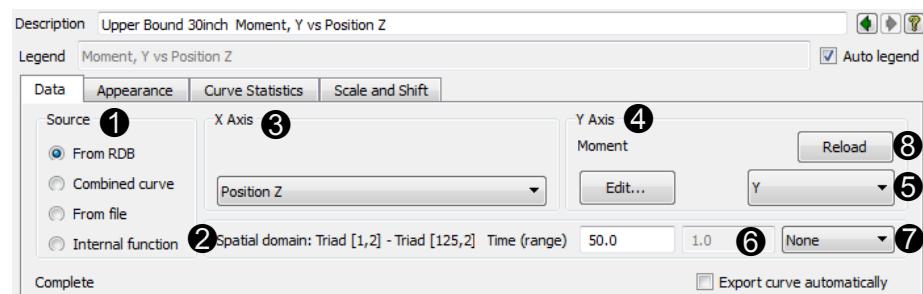


### 9.3.2 Beam diagram curve properties

The properties of a beam diagram graph are identical to those of the regular graphs (see [Section 9.2.3, "Graph properties"](#)), except for that the *Start time* and *Stop time* fields and the *Use time interval* toggle are absent.

The curves in a beam diagram has a slightly different Property Editor panel, compared with the other curves. The *Data* tab reflects the nature of this curve type and is shown below. The other three tabs (*Appearance*, *Curve Statistics* and *Scale and Shift*) are similar as for the regular curves (see [Section 9.2.4, "Curve properties"](#)) whereas the *Fourier Analysis and Differentiation* tab and the *Rainflow and Fatigue* tab both are absent.

9



- ① **Source** – The same choices are available here as for regular curves. However, for *Combined curves*, only other beam diagram curves having identical X-axis definitions may be selected in their definition.
- ② **Spatial domain** – This shows the selected range of objects (Triads or Beams) from which the results plotted by the curve are extracted.

- ③ **X Axis** – Defines the coordinate value used for the x-axis definitions. The choices are *Position X*, *Position Y*, *Position Z* and *Length*. The first three choices indicate the corresponding global position coordinate of the Spatial domain objects, whereas *Length* is a running coordinate along the spatial objects. The latter is useful when the beamstring is curved or not parallel to one of the global coordinate axes.
- ④ **Y Axis** – Defines the result quantity to be plotted. Click the **Edit...** button to open the RDB selector dialog box where you can pick the wanted result (see "[Selecting RDB results](#)" below for details). Alternatively, you may also right-click the curve in the Model Manager *Results* list and then select **Edit Y Axis...**
- ⑤ **Result Operation** – The pull-down menu next to the **Edit...** button lists mathematical operations (such as extracting the x-component or computing the length of a vector) related to the selected result quantity. The content of this menu depends on the chosen result quantity, in the same way as for the regular curves.
- ⑥ **Time (range)** – These fields specify the time or the time range at which the plotted results should be extracted.
- ⑦ **Time operation** – This pull-down menu specifies the mathematical operation that is applied to the result quantity over the selected time range, in order to compute the value to be plotted. The default choice *None* just extracts the results at the time specified in the first of the two *Time (range)* fields. The other options available are *Min*, *Max*, *Abs Max*, *Mean* and *RMS* (root-mean-square).
- ⑧ **Reload** – This button activates a manual reload of the data to be plotted. If you are viewing a beam diagram graph while the dynamics solver is running you need to click on this button to update the plot when new results are available. Unlike the regular curves plotting time histories, beam diagram curves are not updated automatically.

### Selecting RDB results

To select the results to plot in a beam diagram curve, complete the following steps:

1. In the *Results* list of the Model Manager panel, select the Beam diagram curve that you want to edit. Its properties are displayed in the Property Editor panel.
2. In the Property Editor panel, click the **Edit...** button for the y-axis, or right-click on a curve in the Model Manager *Results* list and select **Edit Y Axis...** The RDB selector dialog box (see "[Selecting RDB results](#)" in [Section 9.2.4, "Curve properties"](#)) is then opened.

3. Select a mechanism object from the *Existing Results* list or the *Modeler* view that represent the starting point of the beam diagram curve. Only *Triad* and *Beam* elements may be selected. Selection of any other object is ignored.



**NOTE:** If the mechanism analysis have been performed, variables for the selected object are listed in the Existing Results list. If you have not yet performed an analysis, variables are listed in the Possible Results list.

4. Select a variable to be used as the y-coordinates from either the *Existing Results* or *Possible Results* lists, and click **OK** to close the panel or **Apply** to continue variable selection.

The *Triad* or *Beam* that you select in Step 3 above needs to be at the end of a beam string, i.e., the *Triad* can only be attached to one *Beam* object, or the *Beam* (if selected) can only have a neighboring *Beam* in one of its ends. If these conditions are not fulfilled, the selection is rejected and the curve does not appear in the graph view.

When a valid object selection has been made, the beam topology is traverses automatically to determine the *Spatial domain* objects (see bullet ② above). The traversal is continued until the opposite end of the beam string, or until a *Triad* connected to three or more *Beams* is encountered.



**NOTE:** If a beam string is connected to another (but only one) beam string via a single-master joint (Rigid, Revolute, Ball or Free joint), the beam diagram is continued across that joint. When plotting such beam diagram curves, you may see discontinuities in the plot reflecting the location of that joint.



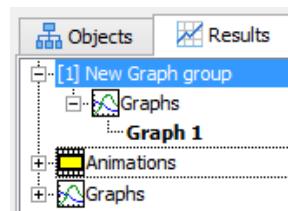
**CAUTION:** Using Length as the x-axis definition (see bullet ③ above) is not recommended if the beam string is interrupted by a joint, because the length coordinate (which is calculated based on updated coordinate values) will then start from zero again on the other side of the joint. It is then better to select the Position X, Y or Z value that best corresponds to the axis direction of the beamstring.

## 9.4 Graph groups

In a similar way as sub-assemblies can be used to group mechanism objects (see [Section 5.15, "Sub-assemblies"](#)), **Graph groups** can be created to organize the graphs of a complex model into logic units.



To create a new graph group object, select **New Graph group** from the *Result* menu, or right-click anywhere in the Model Manager *Results* list and select **New Graph group**. A new Graph group containing one empty Graph then appears in the *Results* list of the Model Manager panel (shown at right).

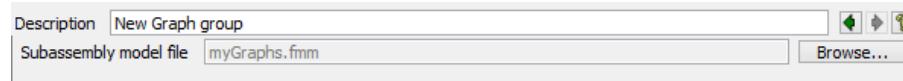


More graphs can be added to the graph group by keeping it selected (highlighted) while creating new Graphs. It is also possible to drag-and-drop result items from the RDB selector dialog box onto a Graph group to create a new graph there with a curve plotting that item. (See "[Graph](#)" and "[Creating curves and graphs by drag and drop](#)" in [Section 9.2.1](#).)



**NOTE:** Only graphs with curves plotting time history results can be organized in graph groups. That is, Beam diagrams can not be created in a graph group object. One may instead say that Beam diagrams constitute a graph group by itself.

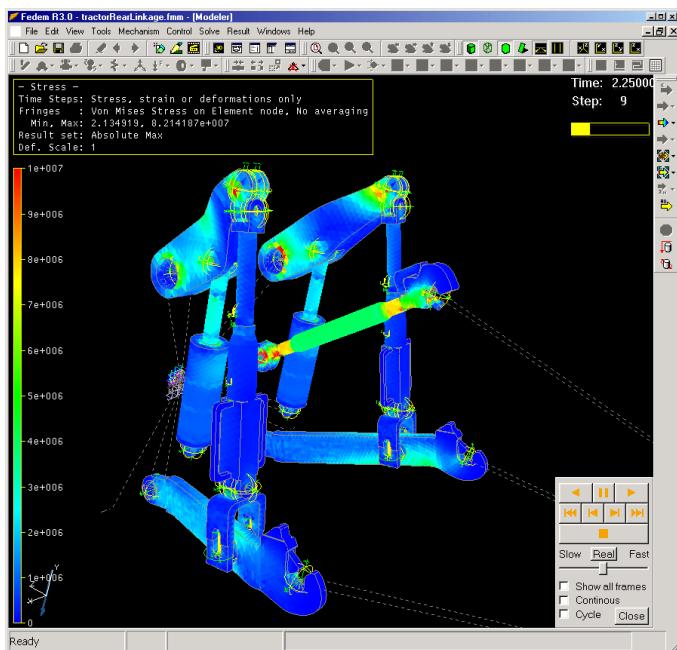
A graph group object is in many ways equivalent to a sub-assembly for mechanism objects, except that it does not contain any positioning data. The Property Editor panel for graph groups thus only contains the *Subassembly model file* field (see below), through which the graph groups can be assigned a file name such that it can be re-used in another model with a similar graph setup.



See [Section 5.15.2, "Sub-assembly properties"](#) for details on how the Subassembly model file field is used.

## 9.5 Animations

Fedem animations are used to visualize the motion or the structural results of your model in an intuitive and life-like fashion. Fedem is able to utilize part motion, part deformation and part color contours to visualize your results and help you understand them.



9

You can create several Animation objects and define different options for each animation. The created animations can then be loaded into the *Modeler* view one at a time.

Animation objects can be set up before or after performing the dynamics simulation and other analyses. If you create a certain type of animation and load it into the *Modeler* view before starting the simulation, you can observe the mechanism motion during the simulation as it is constantly updated (see [Section 8.11, "Interaction during processing"](#)). Otherwise, you can view the entire animation after the simulation is complete.

When an animation is loaded it can be controlled using the Play panel (see [Section 9.6.1, "Play panel"](#)) and the Animation Control dialog box (see [Section 9.6.2, "Animation controls"](#)). The Play panel is displayed in the lower right corner of the *Modeler* view (as shown above) when loaded. The Animation Control dialog box can be activated when needed.

Graphs plotting values vs. time will also be animated in the sense that a bar showing the time of the current animation frame is shown. This time bar is present only as long as the animation is showing a time step.



**NOTE:** You can set up animations at any time and load them; however, nothing will appear unless the appropriate results are present.

### 9.5.1 Managing animations

#### Creating animations

You can create as many animations as you like. It is recommended that you provide descriptive names for your animations (Stress, Strain, or Eigenmodes, and so on) as the description is used in the Model Manager *Results* list to distinguish between the animations.



To create an animation, select **New Animation** on the *Result* menu or right-click in the Model Manager *Results* list and select **New Animation** on the shortcut menu.

#### Loading animations

Once the animation is created, you can show it in the *Modeler* view by performing the following steps:



1. To open the *Modeler* view, click the **Show Modeler** button on the *Windows* tool bar (or *Windows* menu). The *Modeler* view opens in the *Workspace* area.
2. Select the animation in the Model Manager *Results* list.
3. Click the **Load Animation** button in the Property Editor panel, or right-click the animation in the *Results* list and select **Load Animation** on the shortcut menu.

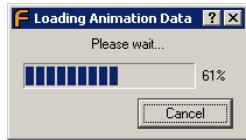


The animation is loaded and both the Contour Legend and Play Panel are displayed in the *Modeler* view (see [Section 9.6.1, "Play panel"](#)).



**TIP:** Changes to animations cannot be updated instantaneously; however, after each change you can reload the animation to include the updates.

Loading an animation with contours and/or deformations for large FE models may take a considerable amount of time. However, it is possible to cancel the animation loading process at any time, by using the **Cancel** button of the progress dialog box that appears while the animation is loading (shown to the right).



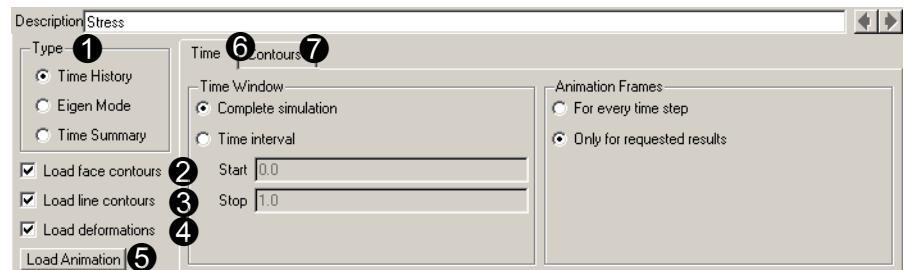
## Closing animations



You can end an animation session at any time by clicking **Close** on the Play Panel or selecting **End Animation Session** from the Result menu.

### 9.5.2 Animation properties

To display the properties of an animation in the Property Editor panel, select the animation in the Model Manager *Results* list. The properties for the animation are displayed in the Property Editor panel as shown below.



- ① **Type** – The animation type determines the main type of results you want to animate. See "[Animation types](#)" below.
- ② **Load face contours** – If enabled, the values specified on the Contours tab are loaded and shown as color contours on the element faces of the mechanism assembly. See also "[Contours tab](#)" below.
- ③ **Load line contours** – This option controls whether or not contours are loaded and displayed on the FE-mesh lines. The values displayed will be the result selected on the Contours tab but always averaged on the nodes. See also "[Contours tab](#)" below.
- ④ **Load deformations** – If enabled, deformation results from the stress or mode shape recovery—depending on the animation type selected—are loaded, if such results are present. See [Section 8.6, "Stress recovery analysis"](#) and [Section 8.7, "Mode shape recovery analysis"](#).
- ⑤ **Load Animation** – If you make changes to the animation properties, you can reload the animation at any time by clicking this button.
- ⑥ **Time tab** – Options to control what part of the time history to load. See "[Time tab](#)" below.
- ⑦ **Contours tab** – Options to control what result values to load and display as color contours. See "[Contours tab](#)" below.



**NOTE:** Element-to-Node averaging is not yet supported, and even if Load line contours is toggled, no contour colors will be shown on the mesh lines when the chosen Result class is Element.

- ④ **Load deformations** – If enabled, deformation results from the stress or mode shape recovery—depending on the animation type selected—are loaded, if such results are present. See [Section 8.6, "Stress recovery analysis"](#) and [Section 8.7, "Mode shape recovery analysis"](#).
- ⑤ **Load Animation** – If you make changes to the animation properties, you can reload the animation at any time by clicking this button.
- ⑥ **Time tab** – Options to control what part of the time history to load. See "[Time tab](#)" below.
- ⑦ **Contours tab** – Options to control what result values to load and display as color contours. See "[Contours tab](#)" below.

## Animation types

The following three animation types are available.

*Time History*– Time history animations are used to animate time history results. This can be the rigid body component of part motion calculated in the dynamics solver, or recovered stress, strain and deformational part of the part motion.

*Eigen Mode*– Eigenmode animations are used to visualize the shape of a system eigenmode at a specific point in time. To provide this visual interpretation, the mode shape is used to create an animation of the mechanism oscillating as if the eigenmode was excited.

The time displayed along with the progress bar shown during an eigenmode animation is the time elapsed during the oscillating motion. The legend text displays the point in time for existence of the animated eigenmode during the dynamics solution.

Unless a mode shape recovery has been run, you can only animate the rigid body component of the eigenmode for each part. Note that the partitioning of the total mode shape into a rigid body approximation plus a deformable component depends on the chosen computational coordinate system of the part (see *Section 4.1* in the *Fedem 7.6 Theory Guide*). In particular, if the center of rotation for a mode is far from the origin of the part coordinate system, the deformable component may be misleading. However, the sum of the deformable and rigid body components will always be correct.

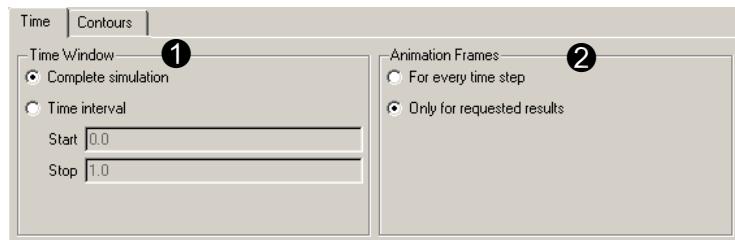
*Time Summary* – Time summary animations are used to show accumulated structural results within a specified time interval. These results are produced by running the Strain Coat Recovery Summary. Examples of such results are the maximal principal stress reached within a time interval, or damage accumulated within the same interval.

Usually only one frame with accumulated results from the complete simulation time are produced and displayed. It is, however, also possible to show an animation of how the results are accumulated by running the Strain Coat Recovery Summary several times with different stop times.

### Time tab

If you selected *Time History* as the animation type in the Property Editor panel , you can choose to show the entire simulation (default) or a specific time interval. To change the interval, you must specify the

properties on the *Time* tab (shown below) in the Property Editor panel, before loading the animation. The *Time* tab is not available for *Eigen Mode* and *Time Summary* animations.



- ① **Time Window** – You can choose to display the entire duration of the simulation or a specific time interval. If you select **Time Interval**, enter values for the interval's *Start* and *Stop* time.
- ② **Animation Frames** – These options are useful if you specified different time steps for the dynamics simulation and the stress recovery calculation (see [Section 8.6.1, "Stress recovery options"](#)). Enabling **For every time step** loads deformation results and color contour values for all the time steps calculated within the time window specified.

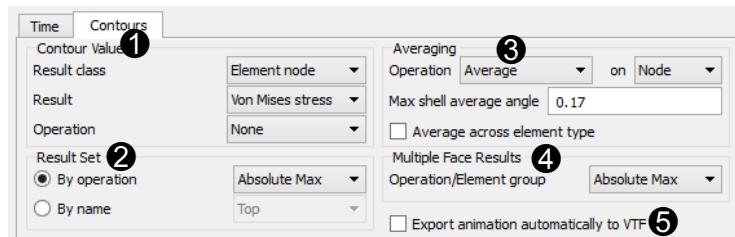


**NOTE:** Enabling **For every time step** shows continuous motion; however, the stress color contours may flash on and off if the calculation intervals are different. If you select **Only for requested results**, then only intervals shown are those at which the stress is calculated; both stress contours and motion appear continuous.

9

### Contours tab

If you selected *Load face contours* and/or *Load line contours* in the Property Editor panel, you can display color contours on the mechanism assembly during the animation (for *Time History* and *Time Summary* animations only). You must then specify the values to be used for the color contours on the *Contours* tab (shown below) in the Property Editor panel, before loading the animation. The *Contours* tab is not available for *Eigen Mode* animation.



① **Contour Value** – These options are used to select the results that will be displayed as color contours.

- *Result class* – this option allows selection of the FE entities for which you want to show color contours. You can select either nodes, elements (one single result for each element), or element nodes (one result for each node within an element). This selection controls the options for the *Result* and *Value* settings (see below).
- *Result* – this option allows you to specify which type of result to display in the animation: stress, strain, deformation, etc. The options available depend on the *Result class* setting, and on the actual results currently available in the results database.
- *Operation* – this option allows you to specify the scalar value to extract from the selected result, such as component selection, von Mises, principle values, and so on. The operations available depend on both the *Result class* and *Result* settings.



**NOTE:** There are two ways of animating some stress and strain measures. E.g. a von Mises stress animation may be set either by choosing "Result - Stress" and "Value - von Mises", or by choosing "Result - von Mises" directly. The reason for this is the option to recover the desired derived stress/strain measures directly (see [Section 8.6.1, "Stress recovery options"](#)). In the above example, if the entire stress tensor was recovered, the first animation definition would be correct. If only the von Mises stress measure was recovered, the second definition should be used. Choosing the wrong definition would leave an empty animation.

② **Result Set** – These options allow you to select one of the named result sets for the *Result class* (node, element, or element node) you selected. This setting is used to distinguish between multiple results for the specified FE entity, including results from different layers within an element (for example, top and bottom of a shell element), or multiple deformation results on a node (for example, mode shape deformations for different eigenmodes).

- *By operation* – enabling this option consolidates the entire result set of the same type using the selected operation. The options include *Average*, *Maximum*, *Minimum*, *Max Difference*, and so on.
- *By name* – enabling this option allows you to load and display only the result sets with the specified name. Complete parts without such a result set will remain unchanged. Elements of a part without the result set will be rendered gray.

③ **Averaging** – The averaging options allow you to control how a continuous contour plot is obtained when the underlying results are

discontinuous across the finite elements. They are available only when the selected *Result class* is *Element node* (see bullet ① above).

- *Operation* – allows you to select the averaging operation to use (*Average*, *Max*, *Min*, etc.), or select *None* if averaging is not wanted
- *on* – averaging on *Node* calculates a single value for each node in the FE model, based on the selected *Operation* and the values specified in the *Contour Value* settings. Averaging on *Element* calculates a single value for each element in the same way.
- *Max shell average angle* – you can specify an angle above which the averaging between shells stops. The angle is used as a tolerance when comparing the globalized coordinate systems of the elements in question.
- *Average across element type* – if enabled, Fedem averages the contour values across element type borders.

- ④ *Multiple Face Results* – This option enables you to specify the averaging behavior across elements that interface on a surface (such as stacking of elements with the same size and shape but different thicknesses). You can either specify an averaging operation, or select a named element group containing the result to be shown.



**TIP:** Contour data is only loaded for visible elements. This means that you can select which of the multiple face results you want to show by hiding appropriate elements.

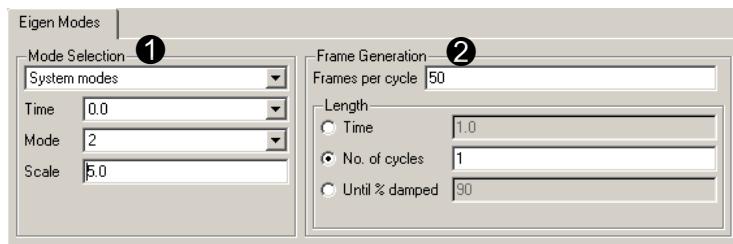
- ⑤ *Export animation automatically to VTF* – When this toggle is enabled, the selected animation will be exported to a VTF-file after a batch stress recovery (see [Section 8.14.1, "Batch solving through the User Interface"](#)) if the command-line options `-exportAnimation` is specified. The name on the VTF-file derived from the *Description* of the animation.

9

### Eigen Modes tab

If you selected *Eigen Mode* as the animation type in the Property Editor panel, you can animate the rigid body mode shapes calculated by Fedem while solving the dynamics. If the Mode Shape Recovery analysis is performed, it is also possible to include part deformations associated with the eigenmode in the animation. You may also animate part mode shapes computed during the FE model reduction if the *Expand mode shape* toggle in the [Reduction Options tab](#) was on (see [Section 5.1.5, "Part properties"](#)).

To animate mode shapes, you must specify the settings on the *Eigen Modes* tab (shown below) in the Property Editor panel, before loading the animation. This tab is not available for *Time History* and *Time Summary* animations.



- ① Mode Selection** – These options allow you to select the mode type, time step or part, mode number, and a scale factor for the animation:
- Select between *System modes*, *Component modes of part* and *Free-free modes of reduced part* from the first pull-down menu.
  - *Time* – If you selected *System modes*, this pull-down menu lists the times at which an Eigenmode solution is specified in the Dynamics Solver Setup dialog box (see "[Eigenmode tab](#)" in [Section 8.5.2](#)).
  - *Part* – If you selected either *Component modes of part* or *Free-free modes of reduced part*, this pull-down menu lists all parts in the model. It also have an entry (*All parts*) to enable simultaneous part mode shape animations for all parts. (See [Section 8.3.6](#), "[Visualization of eigenmode shapes from the model reduction](#)" to learn more on such mode shape visualization.)
  - *Mode* – This pull-down menu shows either the modes specified in the Dynamics Solver Setup dialog box (see "[Eigenmode tab](#)" in [Section 8.5.2](#)), the component mode numbers (see "[Reduction Options tab](#)" in [Section 5.1.5](#)) or the free-free mode numbers for the reduced part, depending on the selected mode type in the first pull-down menu in the *Mode Selection* frame.
  - *Scale* – You can specify a scale factor to exaggerate the mode shapes during the animation.



**Note:** The default Scale setting (1.0) implies that the maximum amplitude of the shape is equal to the length scale used to model the mechanism. Therefore, if the length-span of the model is not 1.0, it is necessary to adjust the Scale option in order to obtain an appropriate deformation scale.

- ② Frame Generation** – These options allow you to set the number of animation frames and the duration of the animation:
- *Frames per cycle* – Entering a higher value increases the continuity of the animated motion.
  - *Length* – You can specify either a *Time*, *No. of cycles*, or *Until % damped* to limit the duration of the animation. These fields are available only for *System modes*. For *Component modes of part* and *Free-free modes of reduced part*, the duration of the animation will always be equal one full cycle.



**NOTE:** The Until % Damped option is relevant only if a damped eigenmode solution was performed (see "Eigenmode tab" in Section 8.5.2).



**NOTE:** When animating eigenmode shapes, the time that runs in the upper right corner of the Modeler view is in the range  $[0, n*T]$  where  $T$  is the period ( $= 1/\text{eigen frequency}$ ) and  $n$  is the number of cycles that are animated (usually  $n=1$  for free vibrations and  $n>1$  for damped vibrations).

### 9.5.3 Available animation results

The results available for animation depend on the solvers that have been run and their settings. For an indication of which results that could be available, use the Result File Browser dialog box (see [Section 10.2, "Result File Browser"](#)).



**TIP:** The heading of the .frs files (Fedem Result File) is readable and could give valuable information in some cases.

If an animation is loaded without the contour results in question being present, the legend text will display a question mark for the max and min values, and no contour colors will appear.

Most of the results that can be viewed in a contour animation are produced during the Stress Recovery analysis. All *Element Node* and *Node* results (selected in the *Result Class* pull-down menu) are currently produced by the Stress Recovery analysis. All *Element* results, however, are produced by the Strain Coat Recovery Summary analysis, and include the following quantities:

- Max principal stress
- Max principal strain
- Max shear stress
- Max shear strain
- Max von Mises stress

- Max von Mises strain
- Maximum stress range
- Maximum strain range
- Mean biaxiality
- Biaxiality standard deviation
- Most popular angle
- Angle Spread
- Damage
- Life (repeats)
- Life (equnits)

### Interpretation of shear strains in contour plots

The stress- or strain state at a point in the model can be represented as a second-order tensor in a mathematical setting, i.e., in 2D we have

$$\sigma = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} \\ \sigma_{yx} & \sigma_{yy} \end{bmatrix} \quad \text{and} \quad \varepsilon = \begin{bmatrix} \varepsilon_{xx} & \varepsilon_{xy} \\ \varepsilon_{yx} & \varepsilon_{yy} \end{bmatrix}$$

where equilibrium in angular momentum requires  $\sigma_{xy} = \sigma_{yx}$ . The strain components are defined as  $\varepsilon_{ij} = (u_{i,j} + u_{j,i})/2$ , where  $u$  is deformation and the indices  $i$  and  $j$  both run through  $x$  and  $y$  (or  $x$ ,  $y$  and  $z$  in 3D). The advantage of this representation is that computation of von Mises and principal quantities can be performed using the same piece of code for stress and strain thereby making the implementation more general and robust.

An alternative representation (more common to engineers) is by means of the stress and strain vectors, i.e., in 2D we have

$$\sigma = \begin{bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{xy} \end{bmatrix} \quad \text{and} \quad \varepsilon = \begin{bmatrix} \varepsilon_{xx} \\ \varepsilon_{yx} \\ \gamma_{xy} \end{bmatrix}$$

where  $\gamma_{xy} = \varepsilon_{xy} + \varepsilon_{yx} = u_{x,y} + u_{y,x}$  and thus  $\gamma_{xy} = 2\varepsilon_{xy}$ .

The results from a Stress Recovery analysis that may be visualized in an animation are based on the tensorial strain representation, i.e., the shear strain components (including Max Shear) are equivalent to the  $\epsilon_{xy}$ -term, and not  $\gamma_{xy}$ . However, the Max Shear strain quantity computed from the Strain Coat Recovery Summary analysis is currently based on the vector representation so that it will be twice as large as the equivalent quantity from the stress recovery analysis. It is important to be aware of this distinction when using that particular result quantity.

### Stress and strain range quantities

The maximum stress- and strain range quantities computed in the Strain Coat Recovery are derived from the principal value history at each point. This computation is performed in a similar manner as for the *Most popular angle* and *Angle Spread* quantities. That is, the directions of the maximum (and minimum) principal values, in terms of the in-plane angle from the x-axis of the stress coordinate system, are used to divide the principal stress states into a discrete number of "bins" for each stress point. The principal stress values at a certain time is then assigned to one such bin depending on its angle, and the range for each such bin is computed as the difference between the highest and the lowest value over the whole time history. The range value presented in the *Time Summary* animations is then selected from the bin having the largest computed range value.

#### 9.5.4 Performance of animation loading

9

It is important to be aware that the performance of the animation loading is highly dependent upon the animation settings, especially when working with large FE-models ( $> 30\,000$  elements).

Memory usage is the most crucial point, and can be reduced to 1/3 when using the better options mentioned below.

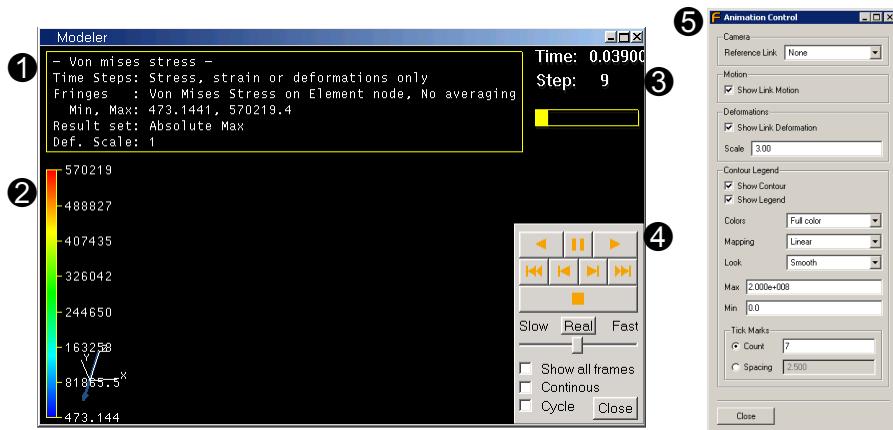
The significant parameters are in order of decreasing importance:

1. *Result Set* selection is *By operation* or *By name*. (*By name* is better.)
2. *Load face contours* and/or *Load line contours* are toggled. (*Only load face contours* is better.)
3. *Averaging* is turned on or off. (*No averaging* is better.)

The number of elements visible when the animation is loaded is also important because result values will only be loaded for visible element faces. This means that you can load contour data for one group at a time if loading data for the complete part is too memory and time consuming.

## 9.6 Viewing animations

When an animation is loaded, you can use several tools to view and explore the data you have loaded. The *Modeler* view will display some additional features and the Animation Control dialog box will be made available. These views are shown in the figure below.

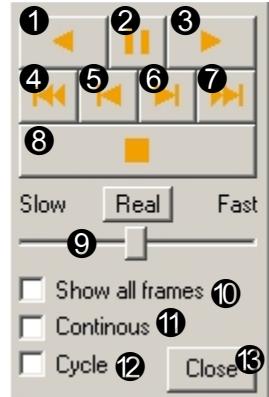


- ① *Legend text* – Displays information regarding the loaded animation
- ② *Legend bar* – Indicates the color assigned to each contour value.
- ③ *Time step information* – Displays the time and the time step number for the current frame along with a progress bar.
- ④ *Play panel* – This panel is used to control how the animation is run.
- ⑤ *Animation Control* – This dialog box features options to control several aspects of the animation, including deformation scale and contour legend domain.

### 9.6.1 Play panel

The Play panel (shown below) is displayed in the lower-right corner of the *Modeler* view when an animation is loaded.

- ① Reverse Play
- ② Pause Play
- ③ Forward Play
- ④ Beginning
- ⑤ Rewind by Frame
- ⑥ Forward by Frame
- ⑦ End
- ⑧ Stop Play
- ⑨ Speed slider – you can adjust the speed of the animation by moving the slider to the right or left. You can also click the **Real** button to return the speed to “real time”.



**NOTE:** At faster speeds, Fedem may skip frames in order to maintain animation speed at the level you have specified.

- ⑩ Show all frames – this option forces Fedem to show all the frames loaded, ignoring the Speed setting if necessary.
- ⑪ Continuous – this option allows you to play the animation repeatedly until you click the *Stop Play* button.
- ⑫ Cycle – enables playing the animation in *Forward Play* mode until it reaches the end, then plays it in *Reverse Play* mode.
- ⑬ Close – This button closes the Play panel and ends the current animation session.



**NOTE:** The Play panel does not appear if the animation consists of a single frame (e.g., Time Summary animations with results from a Strain Coat Recovery). To close such animations, you have to select **End Animation Session** from the Results menu, or type **Ctrl+X**.

## 9.6.2 Animation controls

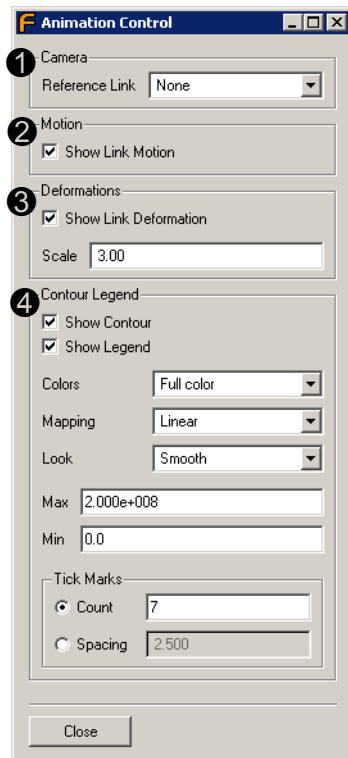


Once an animation is loaded, you can open the Animation Control dialog box by selecting **Show Animation Controls...** on the *Tools* menu. The Animation Control dialog box is opened as shown to the right. Changes made to settings here are instantaneously applied to the animation.

- ① *Camera* – enables selection of reference part for the camera movement. See also "["Camera reference part"](#)" below.
- ② *Motion* – enables display of rigid body motion of the mechanism.
- ③ *Deformations* – enables display of part deformations scaled (exaggerated) by the *Scale* factor you specify.
- ④ *Contour Legend* – these options enable you to customize the Contour Legend, and how the result numbers are converted to colors:
  - *Show Contour* – enables the display of contour contours on the mechanism during the animation.
  - *Show Legend* – enables display of the Contour Legend bar in the *Modeler* view.
  - *Colors* – the type of Color mapping used for the color contour, e.g. Full Color or Red Blue. See also "["Color mappings"](#)" in [Section 9.6.3](#).
  - *Mapping* – *Linear* divides color increments linearly and *Log10* divides color increments on a logarithmic scale.
  - *Look* – either *Smooth* or *Discrete*.



**NOTE:** This scale does not apply to the eigenmode deformations.



**NOTE:** Discrete contours should only be selected when each face in the model has one single color, for example when showing single element results or results averaged on elements.

- *Max/Min* – you can set a maximum and minimum value to show on the Contour Legend. See also "[Contour value domain control](#)" in [Section 9.6.3](#).
- *Tick Marks* – you can select *Count* (number of ticks) or *Spacing* (difference between each tick mark) and specify a value to change the tick marks on the legend. See also "[Tick marks](#)" in [Section 9.6.3](#).

### **Camera reference part**

When viewing an animation, it is possible to make the camera follow the motion of a part in the model. This is useful if some part of your model moves far during the simulation. To do this, selecting the part to follow in the *Reference Part* pull-down menu in the Animation Control dialog box.

### **9.6.3 Contour legend control**

#### **Color mappings**

The drop-down menu labeled *Colors* in the Animation Control dialog box enables the selection of different color mappings for the contour values. The different color mappings map the contour values differently. This concerns values above, within and below the legend domain. They also show undefined results differently.

There are currently four color mappings available:

**Full Color** – This is the default mapping, and is a common way to display structural results.

Values	Color
Above legend domain	Red
Within legend domain	Blue - Cyan - Green -Yellow - Orange - Red
Below legend domain	Blue
Undefined	Gray

**Full Color B/W Limits** – This is a mapping used to show what is within and outside the legend domain.

Values	Color
Above legend domain	White
Within legend domain	Blue - Cyan - Green -Yellow - Orange - Red
Below legend domain	Black
Undefined	Gray

**Full Color Clipped Limits** – This is a mapping that is useful when you want a quick overview of the few elements in a big complex model with results within a certain (narrow) value domain. It renders all elements outside the domain transparent, i.e., the elements within the domain will be the only ones visible.

Values	Color
Above legend domain	Invisible
Within legend domain	Blue - Cyan - Green -Yellow - Orange - Red
Below legend domain	Invisible
Undefined	Invisible

**Red Blue** – This is a mapping useful when you want the interpolation of colors within one element-face to be strictly consistent with the contour legend. Because of limitations in the 3D graphics hardware, interpolations of colors within one face will be linear. That makes the normal full color mapping inconsistent within one face if the different nodes in the element-face have values that map to colors separated by one or more colors in the legend domain.

Values	Color
Above legend domain	Red
Within legend domain	Blue - Red
Below legend domain	Blue
Undefined	Gray

### Contour value domain control

The *Max* and *Min* fields in the Animation Control dialog box are used to control the domain of the color legend. The default value is *0.0* for both. In this case Fedem uses the maximum and minimum values of the loaded contour results.

The *Max/Min* fields can also be used to “flip” the legend scale in order to show the least values as “red” or “worst”. To do this, simply enter the worst value in the field labeled *Max*, and the best value in the field labeled *Min*, ignoring the relative size of the numbers.

When using a logarithmic mapping of the contour values, make sure that both the max and min values are above zero. If not, Fedem will not be able to calculate valid contour colors ( $\log(x)$  does not exist when  $x \leq 0$ ).

### Tick marks

There are two ways to control the tick marks on the color legend. Either by setting the number of tick marks wanted, or by setting the spacing between the tick marks.

When selecting Count, Fedem will distribute the given number of tick marks evenly in the value domain.

When selecting Spacing, Fedem will set tick marks with the given spacing between them, starting at the first complete multiple of the spacing value. When using a logarithmic mapping, the supplied value is multiplied with the decade in question.



**TIP:** When using logarithmic mapping, choose Spacing as tick mark distribution, and either 1, 2.5, 5 or 10 as spacing value.

### Interpreting fatigue results

When calculating fatigue results, some of the elements are rejected because they have near infinite life/zero damage. These elements are excluded from the max/min calculations done by Fedem when reading the results. Fedem assigns them a reference value that make them show up as little damage/ high life when using the ordinary Full Color legend mapping. If “Full Color B/W limits” are used, those elements will show up as Black (below legend domain) if plotting damage, or White (above legend domain) if plotting life. If using “Full Color Clipped Limits” all these elements will be removed from the display, leaving the interesting elements in the display.

The reference values Fedem uses are:

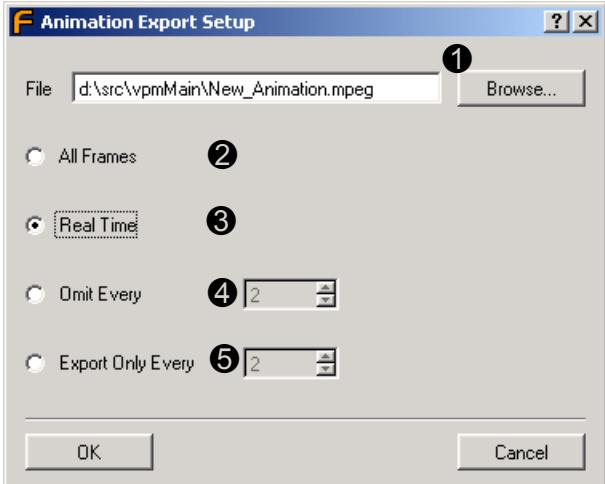
- Damage: 1e-20
- Log Damage: -20
- All life plots (repeats/equunits/Log): 1e20

#### 9.6.4 Exporting animations

The loaded animation can be exported to the mpeg-1, mpeg-2 and avi formats for viewing in an external viewer. Avi export is available on Windows only.

The animation you have loaded forms the basis for the exported animation. Through a simple dialog box, you will be able to control the speed of the animation.

To start export, go to the **File** menu, select **Export** and then **Export Animation...** The Animation Export Setup dialog box (shown below) will then be opened:

- ① **File:** Either type in the file name you want to export to, or click the **Browse...** button to open a dialog box. If you type the file name manually, the given extension will decide which format the animation will be exported to. The default is to save the animation to model file root, with same name as in animation's description. The default file format is mpeg-1.
- 
- ② **All Frames**  
③ **Real Time**  
④ **Omit Every**  
⑤ **Export Only Every**

You can choose between the following operations:

- 
- ② *All Frames*. The animation will be exported "as is". The speed of the exported animation will depend solely on the simulation's time step size.
  - ③ *Real Time*. The animation will be exported so that one second movie time corresponds to one second of simulation time. Note that the animation's frame rate is 30 Hz, so this will be an approximation.
  - ④ *Omit Every*. Every *n*th frame will be omitted when exported.
  - ⑤ *Export Only Every*. Most useful when you have solved using small time steps, and want finer control over what you export.

### Hints and tricks

When exporting to mpeg, the *mpeg-1* format produces smaller files than the *mpeg-2* format, and the quality is approximately the same. For best result, use black background. It seems that the MS Windows Media Player has problems playing the exported mpesgs on some machines. If you discover this, you can try to play the animations with the Elecard mpeg player instead (<http://www.elecard.com>). Also note that the Windows Media Player has a size limit of 720x480 pixels.

If you are working on Windows, we recommend exporting to *avi* format. This export is faster and also produces better quality. When you export to *avi*, a dialog box will pop up where you can set up which codec to use and settings for this codec. Be aware that some codecs may not work, this is dependent on your system configuration. The codecs that seem to have the best success rate is *MS Video 1*, *Cinepak by Radius*, and the *DivX codec*. If you don't have a DivX codec installed on your machine, you can get it from the DivX home page (<http://www.divx.com>).



# Chapter 10 Managing Results

A Fedem simulation can generate large amounts of data. This chapter explains the concept of the Fedem *Results Data Base* (RDB), how to manage the data files and some information on how Fedem stores and handles the data.

Sections in this chapter address the following topics:

- [Model and Result file handling](#)
- [Result File Browser](#)
- [RDB directory structure](#)

## 10.1 Model and Result file handling

A Fedem model consists of the mechanism model file, the FE-model files and all the generated result data files. The generated results can be divided into three main groups:

- FE-model reduction data
- Dynamics response data
- FE-recovery data

Normally all these groups can be looked upon as one single model even though it is spread across several files and directories. Fedem keeps track of which files that are part of your saved model and which are not. Because the solvers write their data directly to disk while solving, Fedem also tracks what files are part of your modified and unsaved model, and can thus tell which files belong to the saved and the unsaved modified version of your model.

### 10.1.1 Discarding unsaved changes

When you open an existing model, the original result files present on disk are preserved, even if you delete results, change the model, run the solvers, etc., as long as you do not save your model. If you exit without saving, all the original data remains unchanged and the result database will be restored to the state of the last save.

The initial data is deleted or overwritten only when the model is saved.

Consequently, to be sure that the result files known by Fedem are those (and only those) present on disk, the model has to be saved first.



**NOTE:** If you use the Result File Browser to delete result files, they will be physically removed from disc instantly. Their removal will not be delayed until the first save. This is done to actually free disk space as you delete the files, and not when you save your model.

### 10.1.2 Saving a model

When the model is saved, the model file is updated with the current information from memory, including information on the contents and status of the results directory. In addition, all changed parts are saved in the FE model repository (default: `link_DB/` directory) and the obsolete files in the results database are deleted.

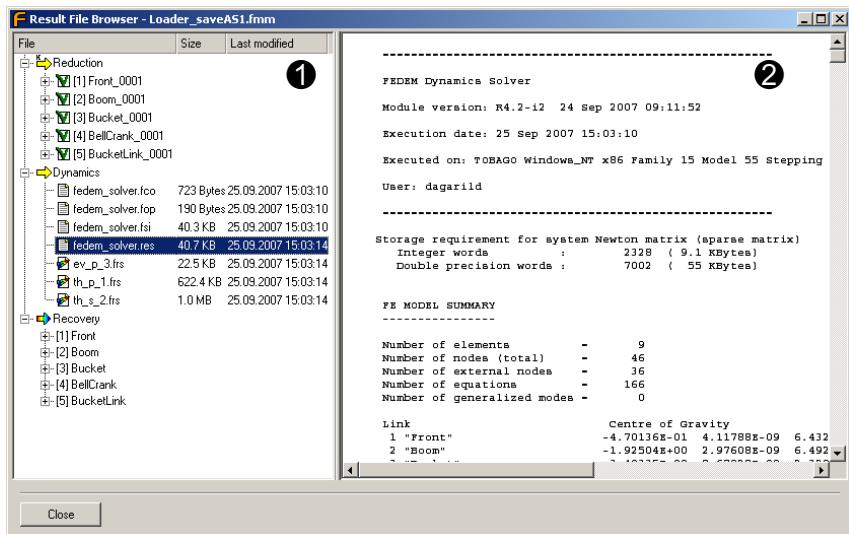
## 10.2 Result File Browser

The Result File Browser is a dialog box that can be used to view and manipulate files created by and used by the various Fedem modules. It offers features such as enable and disable result files, and delete individual results, or classes of results. It responds dynamically to any changes in the results data base, and is kept up-to-date at all times.

### 10.2.1 The Result File Browser dialog



To open the Result File Browser dialog box, select **Result File Browser...** from the *Result* menu, or from the *Windows* tool bar.



10

- ① The *File list* - Lists all relevant files from the Reduction, Dynamics solver and Recovery processes.
- ② The *Info view* - Displays information about the selected file.

### The File list

The list is ordered “chronologically”, with *Reduction* first, then *Dynamics* and *Recovery*. All directories will list the following file types, if present:

- *frs* - Binary result files
- *res* - Log file for the solver processes
- *fco*, *fop*, *fao*, *fsi* - Input files to the solver processes
- *fmx* - Reducer matrix files

- *fsm* - Internal data structure files
- *fpp, fef* - Fatigue result files

The files are listed with file name, size and time of last modification. Some of the files will also have an associated icon, corresponding to the icon that file type will have in Windows Explorer.

When you run a solver process, the result files from it will appear in the file list immediately after it is created, and it will be continuously updated up until the process finishes.

*Reduction* - This list shows the status and location of the results from the reduction process. The icon in front of each part indicates whether the part has a recognized set of reduced matrices or not. A green hatch indicates that the results are recognized as OK, a red cross indicates failure or that the part hasn't yet been reduced.

*Dynamics* - Shows the result files produced by the Dynamics Solver.

*Recovery* - Displays a list of all the parts in your model, and for each part the result files for that part grouped by the recovery process they were created by.

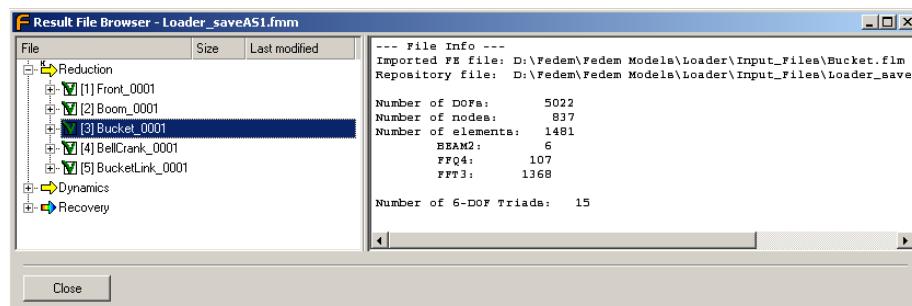


**NOTE:** If you are working on a slow machine and have a lot of results displayed in the file list, continuously updating the list may steal valuable CPU cycles from the solver process. Close the Result File Browser dialog box to disable these updates and improve the overall performance.

### The Info view

When selecting a file in the file list, the file can be viewed in the info view. The plain text files (*fco, fop, fao, fsm* and *res*) are displayed as-is, while selecting an *frs* file will show only the top of the header section.

Selecting a part under *Reduction* will show information about that part.



In addition to the full path to the imported FE data file and the internal repository file, you will here also get a summary of some size parameters of the FE model for the part. This includes the total number of DOFs, the number of nodes and the number elements of each type. The number of triads attached to the part is also indicated.

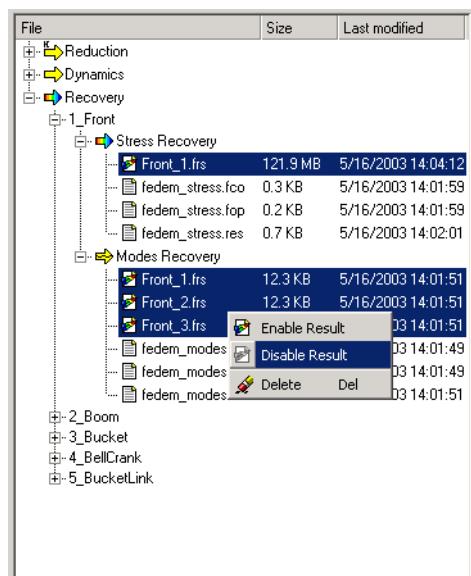
The size information of the FE model is available also before the part is reduced. It is therefore useful for assessing the computational cost of reducing the part. This size information is not shown for Generic Parts, even when a FE model is used for visualizing the part.

### 10.2.2 Result manipulation

The Result File Browser can be used to manipulate the results, both enable/disable results to decrease memory usage, and delete single or multiple result files to save disk space.

#### Disabling and Enabling results

To disable/enable results, select the files you want to enable/disable, right-click, and select either **Enable Results** or **Disable Results** from the menu. The icons of the files immediately changes to reflect the current result state.



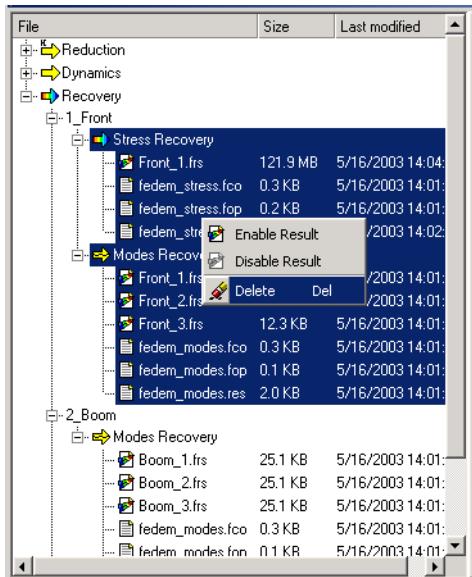
What you actually do when you disable a result file, is to temporarily remove all the result information in the result file from memory, and the results from that file will be unavailable for post-processing (curve plotting and animation). As a result, Fedem consumes less memory. You may re-enable the disabled results any time you wish. The results from this file will then be available for post-processing again.



**NOTE:** Selecting a top level item in the file list will also automatically select any *frs* files located below that item in the list.

### Deleting results

Using the same approach as when *Disabling and Enabling results*, you can also delete individual result files, classes of results, or results on selected parts. Just select the results you want to delete, right-click, and select **Delete** from the menu.



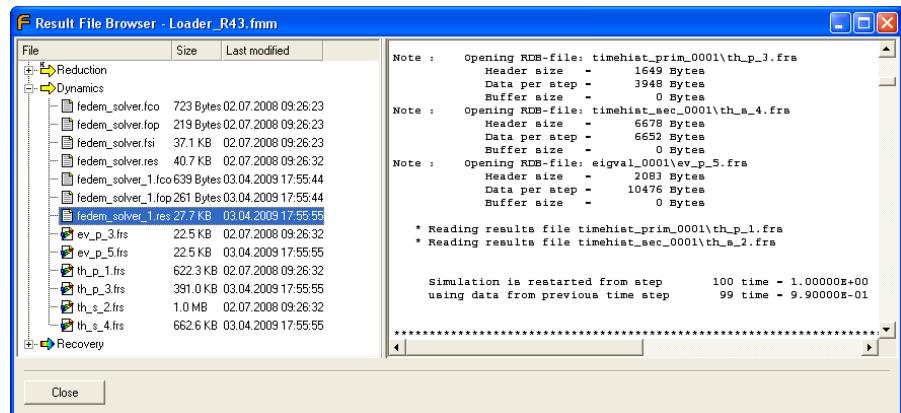
**WARNING!** Deleting the primary time history results (the file named *th\_p\_1.frs*) will also cause all other results to be deleted.



**WARNING!** When you delete results, they are physically removed from disc, and there is no way of getting them back at a later stage. An exception to this is when you delete the primary time history result file. In that particular case the results are not removed from disc until the next time you save your model.

### 10.2.3 Result files from restart simulations

When the Dynamics simulation has been restarted at least once (see "[Time tab](#)" in [Section 8.5.2](#)), the *Dynamics* part of [The File list](#) in the Result File Browser will contain solver option and result files for each individual run. The option files (extension `fop`, `fco` and `fao`) and log-files (`res`) from restart simulations will have a number appended to the base name, indicating the actual restart number (i.e. `fedem_solver_1.res`), whereas the binary result files (`frs`) will have numbers 3 or 4 and beyond, depending on whether eigenmode analysis was activated.



**NOTE:** There will always be only one `fedem_solver.fsi` file in the File list regardless of whether results have been performed or not, because the same file is used by all restart runs. This file only contains model data that is not allowed to change in a restart.

You may Enable/Disable and Delete individual files from restarts in a similar way as the files from the original run (see [Section 10.2.2, "Result manipulation"](#)), in order to control what results should be active for further post-processing and recovery runs. If multiple results are active for a given time set of time steps, only those that were produced latest will be used<sup>1</sup>.

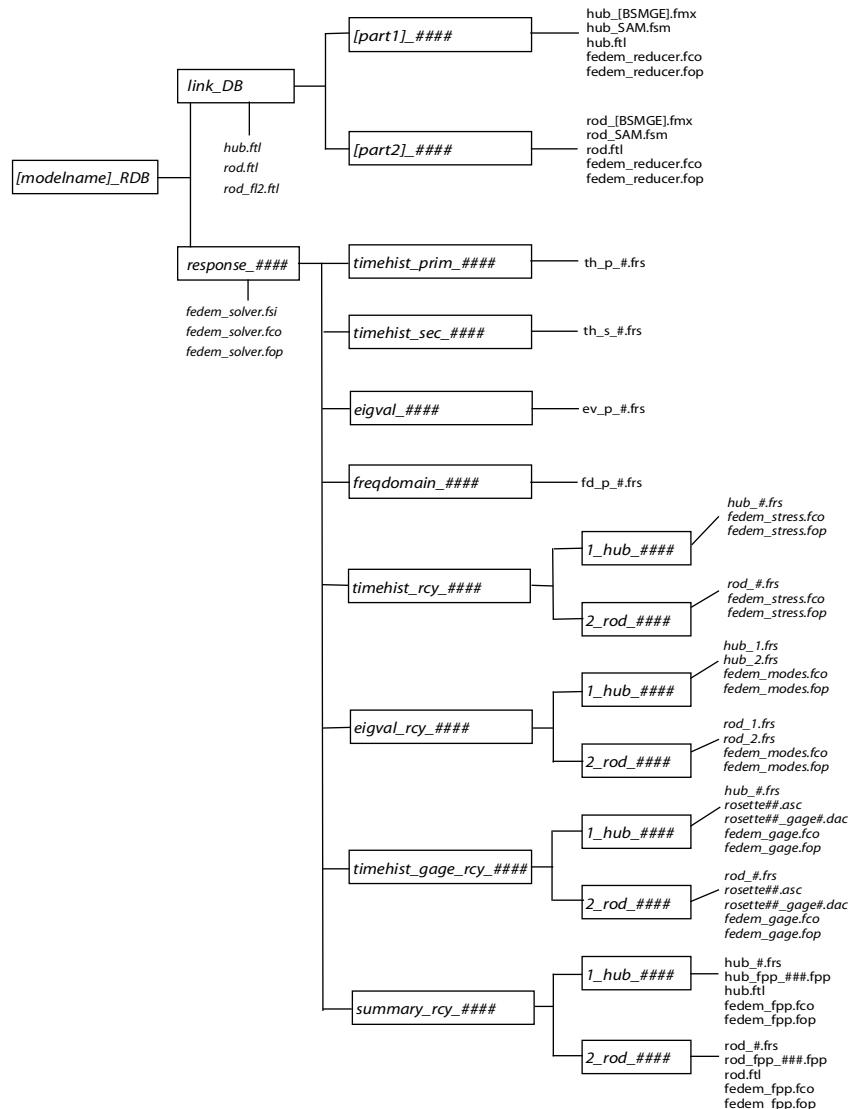


**NOTE:** If deleting a primary time history results file from a restart simulation (e.g. the file `th_p_3.frs` in the view above), all other simulation results are not automatically deleted as well, as happens when the primary result file of the original simulation, `th_p_1.frs`, is deleted).

1. The same is true when post-processing results from recovery simulations that have been rerun several times on the same dynamics simulation results.

## 10.3 RDB directory structure

The map below outlines the RDB directory structure created by Fedem. The hierarchy root is named `[modelname]_RDB`, where `[modelname]` is the name of the current model. For example, the model file `FrontSuspension.fmm` creates a results directory structure under the directory `FrontSuspension_RDB`.



### 10.3.1 Part database

The `link_DB` directory (either specified as default, part-specific or model-specific) contains Fedem `.ftl` files of all unique parts imported into the Fedem model. All `.ftl` files in this directory are stored without external node information. This reduces the total storage requirements.

A set of new subdirectories is created under the `link_DB` directory to store information about already reduced parts. This directory is named `[partname]_####`, where `partname` is the name of the actual part, and `####` represents a configuration number.

Option files for Fedem Reducer are also stored in the `[partname]_####` directories. This enables reduced parts to be moved between result databases.



**NOTE:** When a part- or model-specific repository is used (see [Section 5.1.6, "Using FE model repositories"](#)), the file name conventions apply to the subdirectories of that directory, and not the `link_DB` directory.



**CAUTION:** If you use a file browser to remove unwanted files in the Fedem Results Database directory structure, do not remove the `.ftl` files directly under the `link_DB` directory. Other files will be automatically recreated when a new simulation starts.



**CAUTION:** If you use a file browser to move a Fedem Mechanism Model file (`.fmm`), you must also move the `link_DB` directory to ensure inclusion of the part definitions.

### Model reduction file management

When the model reduction process is started, a new directory for each FE part is created and the files needed by `fedem_reducer` are written to this directory. These files include an `.ftl` file with external node specification, and the options files `.fco`, `.fop` and `.fao` (see [Section B.1, "File types"](#)).

After the reduction, all input files are retained for reference and a possible rerun of the process in batch mode at a later stage.

### 10.3.2 Response directory structure

The `response_####` directory is the entry-point to result files from dynamics simulation and recovery operations.

Option files for the dynamics solver are stored directly in the `response_####` directory. These files can be used to run the solver in batch mode.



**CAUTION:** These files are auto-generated from the main Fedem application. Manually changing the contents of these files and running the solver from the command line creates results that are inconsistent with the definitions in the model file.

All results files in the `response_####` structure are stored in the same format, but are placed in different directories, making separate result types easily distinguishable.

The recovery modules will store their results in a part-wise manner in subdirectories under their main result directory, named `[ID]_[partname]_####`, where `ID` is the part identification number. All option files are also stored in these subdirectories.

The following is a list of the result directories and a description of their contents:

- `timehist_prim_####`: The primary time history result files are named `th_p_.frs` and contain primary response results from the dynamics solver.
- `timehist_sec_####`: The secondary time history result files are named `th_s_.frs` and contain all secondary response results from the dynamics solver.
- `eigval_####`: Calculated eigenvalues and associated eigenvectors from the dynamics simulation are stored in files named `ev_.frs`.
- `freqdomain_####`: Results from the frequency response analysis are stored in files named `fd_p_.frs`. The result variables store here might be in conflict with similar quantities in the `th_p_.frs` files, so make sure only one of them are active in the [Result File Browser](#) when viewing the results.
- `timehist_rcy_####`: Results from the stress recovery process will be stored part-wise in subdirectories. The files will be named `[partname]_.frs`.
- `eigval_rcy_####`: Results from the eigenvalue recovery process will be stored part-wise in subdirectories under this directory, and will be named `[partname]_.frs`.
- `timehist_gage_####`: Results from strain rosette recovery are stored part-wise in subdirectories under this directory. The result files will be named `[partname]_.frs`. Fedem will also create ASCII and DAC files. See [Section 8.8, "Strain rosette analysis"](#).
- `summary_rcy_####`: Results from strain coat recovery are stored part-wise in subdirectories under this directory. The result files are named `[partname]_.frs`.

### Strain Coat Recovery Summary file management

When the Strain Coat Recovery process is started, a new directory called

*response\_####/summary\_rcy\_####/<ID>\_<partname>\_####*

is created, and the files needed by *fedem\_fpp* are written to this directory. These files include an *.ftl* file with Strain Coat elements, and the option files *.fco*, *.fop* and *.fao*.

The Strain Coat elements are of a “non-structural” type and do not affect the model reduction results.



**NOTE:** *The .ftl file written to this directory is not necessarily identical to the corresponding file in the FE model repository, depending on whether the model has been saved since the strain coat elements were created.*



## Appendix A    FE Model Interface

Finite Element (FE) models are generated in external CAE systems and stored in separate files that Fedem refers to as *FE data files*. Fedem stores these files in the Fedem Technology Link format (*.ftl*), and has import filters for the Nastran Bulk Data Format (*.nas* or *.bdf*), the SESAM Input Interface File format, and the older Fedem Link Model format (*.f1m*). This appendix describes these files and their formats.

Sections in this appendix address the following topics:

- [Fedem Technology Link format](#)
- [Nastran Bulk Data File format](#)
- [SESAM Input Interface File format](#)

## A.1 Fedem Technology Link format

In Release 2.5, Fedem Technology introduced the new `.ftl` file format for the definition of FE parts. This format is designed to be flexible, powerful, and extensible for adding new entries in the model. The file is defined in ASCII format and can be easily edited using a text editor.

### A.1.1 Syntax

An `.ftl` file contains a set of identifiers (nodes, elements, control data, and attributes) and parameters expressed with the same overall syntax:

```
identifier{id value1 ... valueN {reference id text}}
```

Name	Description
<code>identifier</code>	Specifies field type (e.g., element type, attribute type).
<code>id</code>	Unique ID for the field (relative to the other fields with the same identifier).
<code>value1 ... valueN</code>	Primary values for the object (can be text, integers, or decimal digits).
<code>references</code>	Additional data or other fields can be referred to using this field.
<code>reference and id</code>	Field reference (field type specified in combination with a valid ID).
<code>text</code>	Can be used as additional information for a field reference or as an optional tag (e.g., a group name)

The following are examples:

```
TET4{4 22 34 12 32 {PMAT 1}}
```

A 4-noded tetrahedron element with ID 4, referring to nodes 22, 34, 12 and 32. The element uses an attribute of type PMAT with ID 1.

```
PMAT{1 2.10e+11 8.00e+10 2.90oe-01 7.82e+03}
```

A material property entry with ID 1 and four decimal numbers describing the different parameters in the material.



**NOTE:** All text between a comment symbol ("#") and the end of the line is ignored.

## A.1.2 Nodes

Nodes are expressed by the following syntax:

*NODE{id state x y z}*

Parameter	Value Type	Description
<i>id</i>	Integer	External node identifier
<i>state</i>	Integer	Internal/external state flag = 0 (an internal node condensed out in the reduction) = 1 (an external node retained in the reduction) < 0 (an internal node with constraints, the value is a binary coding of the fixed DOFs)
<i>x y z</i>	Real	Global nodal coordinates

## A.1.3 Structural elements

Elements are expressed in different ways depending on the element type, using the statements in the table below.

Element statements
<i>BEAM2{id n1 n2 {PMAT pid} {PBEAMSECTION gid} [{PORIENT oid}] [{PBEAMECCENT eid}] [{PBEAMPIN bpid}] [{PEFFLENGTH lid}] [{PNSM nid}]}</i>
<i>BEAM3{id n1 n2 n3 {PMAT pid} {PBEAMSECTION gid} [{PORIENT oid}] [{PORIENT3 oid}] [{PBEAMECCENT eid}] [{PBEAMPIN bpid}] [{PEFFLENGTH lid}] [{PNSM nid}]}</i>
<i>TRI3{id n1 n2 n3 {PMAT pid} {PTHICK gid} [{PNSM nid}]}</i>
<i>TRI6{id n1 n2 ... n6 {PMAT pid} {PTHICK gid} [{PNSM nid}]}</i>
<i>QUAD4{id n1 n2 n3 n4 {PMAT pid} {PTHICK gid} [{PNSM nid}]}</i>
<i>QUAD8{id n1 n2 ... n8 {PMAT pid} {PTHICK gid} [{PNSM nid}]}</i>
<i>TET4{id n1 n2 n3 n4 {PMAT pid}}</i>
<i>TET10{id n1 n2 ... n10 {PMAT pid}}</i>
<i>WEDG6{id n1 n2 ... n6 {PMAT pid}}</i>
<i>WEDG15{id n1 n2 ... n15 {PMAT pid}}</i>
<i>HEX8{id n1 n2 ... n8 {PMAT pid}}</i>
<i>HEX20{id n1 n2 ... n20 {PMAT pid}}</i>

Element statements
BUSH{id n1 n2 [{PBUSHCOEFF bcid}] [{PBUSHECCENT beid}] [{PORIENT oid}   {PCOORDSYS csid}]}
SPRING{id n1 n2 {PSPRING sid}}
RSPRING{id n1 n2 {PSPRING sid}}
CMASS{id n1 [{PMASS mid}]}
RBAR{id n1 n2 {PRBAR rid}}
RGD{id n1 n2 ... nn [{PRGD rid}]}
WAVGM{id n1 n2 ... nn {PWAVGM wid}}



**NOTE:** The identifiers correspond to the element types defined in the Fedem 7.6 Theory Guide, Appendix A, "Finite Element library".



**NOTE:** The square brackets ([]) denote optional parameters. The vertical bar (|) means that either one of the two parameters on each side of it may be specified, but not both.



**NOTE:** All element types listed above may have the reference field {VDETAIL vid} in addition to the property references listed.



**NOTE:** The elements BUSH and CMASS may exist without any associated properties (PBUSHCOEFF and PMASS, respectively) in the .ftl file. Such elements are created automatically by Fedem during modeling, e.g. when a mechanism joint is attached to a slave FE node in a part (see [Section 4.6, "Attaching and detaching elements"](#)). When the Fedem Reducer encounters such property-less elements, some stiffness/mass properties are computed automatically for these elements based on the assembled stiffness/mass matrix of the whole part, such that the element can be regarded as nearly mass-less and rigid. Refer to the Fedem 7.6 Theory Guide, Appendix A, "Finite Element library" for details on this computation.

Parameters for element statements are given in the table below.

Parameter	Description
<i>id</i>	External element identifier
<i>ni</i>	Reference to the <i>i</i> <sup>th</sup> node connected to this element
<i>bcid</i>	Reference to a stiffness coefficient field for bushing elements
<i>beid</i>	Reference to an eccentricity field for bushing elements
<i>bpid</i>	Reference to a pin flag field for beam elements
<i>csid</i>	Reference to a local coordinate system field
<i>eid</i>	Reference to an eccentricity field for beam elements

Parameter	Description
<i>gid</i>	Reference to a geometric property field for beam and shell elements
<i>lid</i>	Reference to an effective length field for beam elements
<i>mid</i>	Reference to a mass property field for concentrated mass elements
<i>nid</i>	Reference to a non-structural mass field for beam and shell elements
<i>oid</i>	Reference to an orientation field for this element
<i>pid</i>	Reference to a material field for this element
<i>rid</i>	Reference to a component numbers field for rigid elements
<i>sid</i>	Reference to a stiffness matrix field for spring elements. For SPRING elements it refers to a PSpring entry with type=1, whereas for RSPring elements it refers to a PSpring entry with type=2.
<i>vid</i>	Reference to a visibility status field for this element
<i>wid</i>	Reference to a weight- and component numbers field for weighted averaged motion elements

#### A.1.4 Properties

The various properties that are used in the structural element expressions have the following syntax:

*PMAT{pid e g v p}*

Parameter	Value Type	Description
<i>pid</i>	Integer	Material property identifier
<i>e</i>	Real	Young's modulus
<i>g</i>	Real	Shear modulus (used by beam elements only)
<i>v</i>	Real	Poisson's ratio
<i>p</i>	Real	Material density

*PBEAMSECTION{gid a iyy izz ixx ky kz cy cz}*

Parameter	Value Type	Description
<i>gid</i>	Integer	Geometric property identifier
<i>a</i>	Real	Cross-sectional area
<i>iyy izz</i>	Real	Moments of inertia about the local y- and z-axes of the beam
<i>ixx</i>	Real	Torsional stiffness parameter
<i>ky kz</i>	Real	Transverse shear reduction factors
<i>cx cz</i>	Real	Local y- and z-coordinates of the shear center of the beam

*PBEAMECCENT{eid ex1 ey1 ez1 ex2 ey2 ez2 [ex3 ey3 ez3]}*

Parameter	Value Type	Description
<i>eid</i>	Integer	Eccentricity vectors identifier
<i>ex1 ey1 ez1</i>	Real	Eccentricity vector at local node 1
<i>ex2 ey2 ez2</i>	Real	Eccentricity vector at local node 2
<i>ex3 ey3 ez3</i>	Real	Eccentricity vector at local node 3

*PEFFLENGTH{lid leff}*

Parameter	Value Type	Description
<i>lid</i>	Integer	Effective length identifier
<i>leff</i>	Real	Effective beam length

*PBEAMPIN{bpid pa pb}*

Parameter	Value Type	Description
<i>bpid</i>	Integer	Beam pin flag identifier
<i>pa</i>	Integer	Local DOFs in end 1 that are released
<i>pb</i>	Integer	Local DOFs in end 2 that are released

*PNSM{nid rho flag}*

Parameter	Value Type	Description
<i>nid</i>	Integer	Non-structural mass identifier
<i>rho</i>	Real	The non-structural mass per unit length if <i>flag</i> =0, and per unit area if <i>flag</i> =1
<i>flag</i>	Integer	Flag indicating if this entry is used by a beam or shell

*PTHICK{gid t}*

Parameter	Value Type	Description
<i>gid</i>	Integer	Geometric property identifier
<i>t</i>	Real	Shell thickness

*PBUSHCOEFF{bcid k1 k2 k3 k4 k5 k6}*

Parameter	Value Type	Description
<i>bcid</i>	Integer	Bushing coefficients identifier
<i>ki</i>	Real	Stiffness coefficients in local directions of the bushing element

*PBUSHECCENT{beid ex ey ez}*

Parameter	Value Type	Description
<i>beid</i>	Integer	Eccentricity vector identifier
<i>ex ey ez</i>	Real	Offset vector from local node 1 to the bushing element location

*PORIENT{oid ox oy oz}*

Parameter	Value Type	Description
<i>oid</i>	Integer	Orientation vector identifier
<i>ox oy oz</i>	Real	Local z-axis of the element

*PORIENT3{oid ox1 oy1 oz1 ox2 oy2 oz2 ox3 oy3 oz3}*

Parameter	Value Type	Description
<i>oid</i>	Integer	Orientation vector identifier
<i>ox1 oy1 oz1</i>	Real	Local z-axis at element node 1
<i>ox2 oy2 oz2</i>	Real	Local z-axis at element node 2
<i>ox3 oy3 oz3</i>	Real	Local z-axis at element node 3

*PCOORDSYS{csid ox oy oz zx zy zz px py pz}*

Parameter	Value Type	Description
<i>csid</i>	Integer	Local coordinate system identifier
<i>ox oy oz</i>	Real	Origin of the local coordinate system
<i>zx zy zz</i>	Real	Local Z-axis in the coordinate system
<i>px py pz</i>	Real	Point in the local XZ-plane

*PSPRING{sid k11 k21 k22 k31 k32 k33 k41 k42 k43 k44 k51  
k52 k53 k54 k55 k61 k62 k63 k64 k65 k66 type}*

Parameter	Value Type	Description
<i>sid</i>	Integer	Spring stiffness matrix identifier
<i>kij</i>	Real	Component (i,j) of the symmetric 6x6 spring stiffness matrix
<i>type</i>	Integer	Spring type flag =1 : Translatory spring =2 : Rotational spring

*PMASS{mid I11 I21 I22 I31 I32 I33 I41 I42 I43 I44 I51 I52  
I53 I54 I55 I61 I62 I63 I64 I65 I66}*

Parameter	Value Type	Description
<i>mid</i>	Integer	Mass property identifier
<i>Iij</i>	Real	Component (i,j) of the symmetric point-mass matrix

*PRBAR*{*rid* *cn1* *cn2* *cm1* *cm2*}

Parameter	Value Type	Description
<i>rid</i>	Integer	Rigid bar property identifier
<i>cn1</i> <i>cn2</i>	Integer	Component numbers of independent DOFs in the part coordinate system for the element at end 1 and 2 respectively
<i>cm1</i> <i>cm2</i>	Integer	Component numbers of dependent DOFs in the part coordinate system assigned by the element at end 1 and 2 respectively

*PRGD*{*rid* *cm*}

Parameter	Value Type	Description
<i>rid</i>	Integer	Rigid body property identifier
<i>cm</i>	Integer	Component numbers of the dependent DOFs in the part coordinate system at all slave nodes of the rigid element

*PWAVGM*{*wid* *rc* *x1* ... *x6* *w11* ... *w1n* *w21* ... *w2n* ...}

Parameter	Value Type	Description
<i>wid</i>	Integer	Weighted averaged motion property identifier
<i>rc</i>	Integer	Component numbers of the dependent DOFs at the slave node of the weighted average motion element.
<i>xi</i>	Integer	Row index into the weighting matrix for local DOF <i>i</i> at the slave node
<i>wij</i>	Real	Weighting factor at master node <i>j</i> for the element, for all slave DOFs whose row index equals <i>i</i>

*VDETAIL*{*vid* *visible*}

Parameter	Value Type	Description
<i>vid</i>	Integer	Visibility status identifier
<i>visible</i>	Integer	Visibility flag (value 0 means invisible element faces whereas 1 means visible element faces)

### A.1.5 Loads

Both concentrated point loads on nodes and distributed surface loads on shell and solid elements are supported. The FTL-syntax is as follows:

```
CFORCE{sid fx fy fz n1 ... n<n>}

CMOMENT{sid mx my mz n1 ... n<n>}

SURFLOAD{sid p1 ... p<n> e1 ... e<n> [{PORIENT oid}]}

FACELOAD{sid p1 ... p<n> e1 f1 ... e<n> f<n> [{PORIENT oid}]}
```

Parameter	Value Type	Description
<i>sid</i>	Integer	Load set identifier
<i>fx fy fz</i>	Real	Global force components
<i>mx my mz</i>	Real	Global torque components
<i>p&lt;i&gt;</i>	Real	Surface force intensity in local node <i>i</i>
<i>n&lt;i&gt;</i>	Integer	Node IDs of load target points
<i>e&lt;i&gt;</i>	Integer	Element ID
<i>f&lt;i&gt;</i>	Integer	Local face number

### A.1.6 Strain Coat Elements

Four strain coat element types are supported, where the type names reflect the number of element nodes. The FTL-syntax is as follows:

```
STRCT3{id n1 n2 n3 {PSTRC pid1} ... {PSTRC pid<n>} {FE eid} }

STRCT6{id n1 n2 ... n6 {PSTRC pid1} ... {PSTRC pid<n>} {FE eid} }

STRCQ4{id n1 n2 n3 n4 {PSTRC pid1} ... {PSTRC pid<n>} {FE eid} }

STRCQ8{id n1 n2 ... n8 {PSTRC pid1} ... {PSTRC pid<n>} {FE eid} }
```

Parameter	Description
<i>id</i>	External element identifier
<i>n&lt;i&gt;</i>	Reference to the <i>i</i> 'th node connected to this element
<i>pid&lt;i&gt;</i>	Reference to the property field of the <i>i</i> 'th calculation point for this element
<i>eid</i>	Reference to the underlying structural FE element

### A.1.7 Strain Coat Properties

The property fields that are referred to by the strain coat element fields have the following syntax:

*PSTRC{pid name {PMAT mid} [{PTHICKREF tid}|{PHEIGHT hid}]} }*

Parameter	Value Type	Description
<i>pid</i>	Integer	Strain coat property identifier
<i>name</i>	String	Result set name to be displayed in the animation UI (one of "Top", "Bottom" or "Basic")
<i>mid</i>	Integer	Reference to a material property field
<i>tid</i>	Integer	Reference to a thickness-relative z-position field
<i>hid</i>	Integer	Reference to an absolute z-position field

*PTHICKREF{tid fact {PTHICK gid}} }*

Parameter	Value Type	Description
<i>tid</i>	Integer	z-position identifier
<i>fact</i>	Real	Location of the calculation point in the thickness direction of a shell as a fraction of the referenced shell thickness, i.e., the z-position is $z = fact * t$ where <i>t</i> is the thickness referenced through the parameter <i>gid</i> .
<i>gid</i>	Integer	Reference to a thickness field

*PHEIGHT{hid h}*

Parameter	Value Type	Description
<i>hid</i>	Integer	<i>z</i>
<i>h</i>	Real	Absolute location of the calculation point in the thickness direction of a shell, i.e., the z-position is $z = h$

### A.1.8 Other identifiers

The following identifiers are used to define element groups:

*GROUP{id e1 e2 ... en {NAME name}}*

Parameter	Value Type	Description
<i>id</i>	Integer	Group identifier
<i>ei</i>	Integer	Element ID of the <i>i</i> <sup>th</sup> element in this group
<i>name</i>	String	Name tag of this group

## A.2 Nastran Bulk Data File format

Fedem supports a wide range of Nastran bulk data entries (see *table below*). For most element types, implicit conversion to a known Fedem element type is performed. Refer to Nastran's Bulk Data File documentation for details about properties and syntax for each entry, and the Fedem 7.6 Theory Guide, Appendix A, "Finite Element Library" for details about the Finite Element library in Fedem.

FE models in *Nastran bulk data file* format can be directly imported as parts into Fedem using the Import Part command (see [Section 5.1.2, "Creating parts by file import"](#)).



**NOTE:** Each FE model to be imported into Fedem must be stored in a separate bulk data file with the Nastran Bulk Data File extension (.nas or .bdf).

Nastran Bulk Data Conversion	Comments
BAROR	
BEAMOR	
CBAR	
CBEAM	Same as CBAR
CBUSH	
CELAS1	
CELAS2	Same as CELAS1
CHEXA	Supports both 8 and 20 nodes
CONM1	
CONM2	Same as CONM1
CONROD	
CORD1C	
CORD1R	
CORD1S	
CORD2C	
CORD2R	
CORD2S	
CPENTA	Supports both 6 and 15 nodes
CQUAD4	

Nastran Bulk Data Conversion	Comments
CQUAD8	
CROD	Same as CONROD
CTETRA	Supports both 4 and 10 nodes
CTRIA3	
CTRIA6	
CWELD	
FORCE	
GRDSET	
GRID	
INCLUDE	
MAT1	See below
MOMENT	
PBAR	
PBARL	See below
PBEAM	
PBEAML	See below
PBUSH	
PELAS	
PLOAD2	
PLOAD4	
PROD	
PSHELL	
PSOLID	
PWELD	
RBAR	
RBE2	
RBE3	
ASET	Automatic definition of external nodes
ASET1	Same as ASET
SPC	See below
SPC1	See below



**NOTE:** For the PBARL/BPEAML entries, Fedem currently supports the following cross section types (see the MSC/Nastran Reference guide for details): "ROD", "TUBE", "BAR", "BOX", "I" and "T". Any other cross section types have to be manually replaced by equivalent PBAR/PBEAM entries.



**NOTE:** The shear modulus in the MAT1 bulk entry is only used by beam elements. If the value on the Nastran file is zero for a MAT1 entry that is used by a beam element, the shear modulus is automatically recomputed from the Young's modulus and Poisson's ratio through the formula  $G = E/(2+2*\nu)$ . However, if  $G=0$  is desired for a beam element, that is still possible by editing the fti-file created in the link\_DB directory when the model is saved.



**NOTE:** If the Poisson's ratio in the MAT1 bulk entry is not given or is outside the valid range [0,0.5>, but the shear modulus is given, the Poisson's ratio will be derived from the Young's modulus and the shear modulus, through the expression  $\nu = E/2G - 1$ , if that yields a value within the valid range.



**NOTE:** The SPC (and SPC1) bulk entry is used to specify fixed DOFs in FE nodes and is an alternative to specify the node as external and constrain it on the system level. Non-zero prescribed values are however treated as zero in Fedem (always fixed).

## A.3 SESAM Input Interface File format

Fedem supports the SESAM input interface file data entries as shown in the table below.

SESAM keyword	Comments
BELFIX	
BLDEP	
BNBCD	Only the Fixed(1) and Free(0) status codes are considered in Fedem. Other values are ignored.
BNMASS	
GBEAMG	
GCOORD	
GECCEN	
GELMNT1	See table below for supported element types
GELREF1	
GELTH	
GSETMEMB	
GUNIVEC	
MGSPRNG	
MISOSEL	
TDMATER	
TDSECT	
TDSETNAM	

The following SESAM element types are supported:

SESAM type name	Equivalent FEDEM type name
BEPS	BEAM2
BEAS	BEAM2
SECB	BEAM3
BTSS	BEAM3 subdivided to two BEAM2
GSPR	RSPRING
CSTA	TRI3
FTRS	TRI3
LQUA	QUAD4
FQUS	QUAD4
ILST	TRI6
SCTS	TRI6
IQQE	QUAD8
SCQS	QUAD8
ITET	TET10
IPRI	WEDGE15
IHEX	HEX20
TETR	TET4
TPRI	WEDGE6
LHEX	HEX8
GMAS	CMASS



## Appendix B    File Types and Usage

This appendix describes the file types used by Fedem and explains which program module use which files.

Sections in this appendix address the following topics:

- [File types](#)
- [File usage for each program module](#)

## B.1 File types

Fedem uses three main types of files: input, intermediate, and results files. Fedem files have an ASCII text or Binary (platform-independent) format, a common tag syntax, and use intuitive naming conventions.

### B.1.1 Input files

Fedem uses several types of input files to import FE models, existing mechanism models, and simulation data as listed in the following table.

Ext.	File description	Content	Format
.fmm	Fedem Mechanism Model file	Mechanism Model and Graph/Animation description	ASCII
.ftl	Fedem Technology Link file	FE model data	ASCII
.ftc	Fedem Technology Cad file	Cad geometry data	ASCII
.nas .bdf	Nastran Bulk Data File	FE model data	ASCII
.FEM	SESAM Input Interface File	FE model data	ASCII
.flm	Fedem Link Model (used by previous Fedem versions)	FE model data	ASCII
.dac	nCode DAC file	Time history data	Binary
.asc .txt	ASCII text file	Time history data (xy-paired coordinates)	ASCII
.rsp .drv .tim	MTS RPC file	Time history data	Binary
.tpf	TNO Tire file	Tire model description	ASCII
.tir	COSIN Tire file	Tire model description	ASCII
.rdf	Road property file	Road description	ASCII

## B.1.2 Intermediate files

Fedem uses the following intermediate files to store analysis options and simulation data from the analyses.

Ext.	File description	Format
.fsi	Fedem Solver Input file	ASCII
.fco	Fedem Calculation Options file <sup>*</sup>	ASCII
.fop	Fedem Output Options file <sup>†</sup>	ASCII
.fao	Fedem Additional Options file	ASCII
.fsm	Fedem SAM data file	Binary
.fmx	Fedem Matrix file	Binary

\* Changes to files of this type affect the calculated response.

† Changes to files of this type do not affect the calculated response, only the amount of output.

## B.1.3 Results files

The following table lists the results file types.

Ext.	File type	Format
.frs	Fedem results database file	Binary
.res	Fedem results output file	ASCII

## B.1.4 Other files

The following table lists all other file types used by Fedem.

Ext.	File type	Format
.fcd	Fedem unit conversion file	ASCII
.fsn	Fedem S-N curves file	ASCII

## B.2 File usage for each program module

The table below is an overview of Fedem file usage.

Fedem application	Uses these file types	Creates these file types
User Interface	FE data files (.ftl, .nas, .bdf, .FEM, .f1m) CAD geometry files (.wrl, .ftc) Time history data files (.asc, .txt, .dac, .rsp, .drv, .tim) – optional Tire description files (.tpf, .tir) – optional Unit conversion file (.fcd) – optional Fedem results files (.frs) S-N curve definition file (.fsn)	Model file (.fmm) FE data files (.ftl) CAD geometry files (.ftc) Solver input files (.fsi) Solver control files (.fco, .fop, .fao)
Fedem Reducer (FE model reduction)	FE data file (.ftl) Solver control files (.fco, .fop, .fao)	Matrix files (stiffness, mass, gravitation, loads, component modes, recovery) (.fmx) Data structure file (.fsm) Result files (.frs) ASCII results (.res)
Fedem Solver (dynamics solver)	Solver input file (.fsi) Solver control files (.fco, .fop, .fao) Matrix files (stiffness, mass, gravitation, loads) (.fmx) Time history data files (.asc, .txt, .dac, .rsp, .drv, .tim) – optional Tire description files (.tpf, .tir) – optional Road description files (.rdf) – optional CAD geometry files (.ftc) – optional Solver result files (.frs) – optional, in case of restart	Fedem results files (time history response, frequency response) (.frs) ASCII results (.res)
Fedem Stress (stress recovery)	Solver input file (.fsi) Solver control files (.fco, .fop, .fao) Matrix files (recovery, component modes) (.fmx) Data structure file (.fsm) FE data file (.ftl) Solver result files (.frs)	Result files (.frs) ASCII results (.res)

Fedem application	Uses these file types	Creates these file types
Fedem Modes (eigenmode recovery)	Solver input file (.fsi) Solver control files (.fco, .fop, .fao) Matrix files (recovery, component modes) (.fmx) Data structure file (.fsm) FE data file (.ftl) Solver result files (.frs)	Result files (.frs) ASCII results (.res)
Fedem Gage (strain rosette recovery)	Solver input files (.fsi) Solver control files (.fco, .fop, .fao) Rosette input files (.ros, .dat) Matrix files (recovery, component modes) (.fmx) Data structure file (.fsm) FE data file (.ftl) Solver results files (.frs)	Result files (.frs) ASCII results (.res, .asc) DAC time history files (.dac)
Fedem FPP (strain coat recovery)	Solver input file (.fsi) Solver control files (.fco, .fop, .fao) Matrix files (recovery, component modes) (.fmx) Data structure file (.fsm) FE data file (.ftl) Solver result files (.frs) S-N curve definition file (.fsn) – optional	Result files (.frs) ASCII results (.res)
Fedem Graph-Exp (curve export utility)	Solver control files (.fco, .fop, .fao) Model file (.fmm) Results files (.frs)	RPC time history data file (.rsp) DAC time history files (.dac) ASCII time history data files (.asc)



## Appendix C Command line options

Each of the Fedem programs may be run manually from a console window or using the **Run** option from the *Start* menu in Windows. To facilitate such batch-execution of the programs, the complete list of command-line options for each solver module is given in this appendix. Any of these options may also be specified in the Additional Solver Options dialog box (see [Section 8.2.3, "Additional solver options"](#)).

The command-line options may contain both upper case and lower case letters. However, the interpretation of the options is case insensitive. For options accepting a numerical (or text string) value, a '=' character may optionally be added between the option and its value. Thus, all the following option specifications are equivalent:

```
-myOption 1.0  
-myoption 1.0  
-MYOPTION 1.0  
-myOption=1.0
```

If you mis-spell or specify a non-existing option, the option is ignored. A warning for each unrecognized option is printed to the console window, or in the [Output List](#) if executed through the user interface.

Sections in this appendix address the following topics:

- [Fedem GUI options](#)
- [Reducer options \(fedem\\_reducer\)](#)
- [Dynamics solver options \(fedem\\_solver\)](#)
- [Stress recovery options \(fedem\\_stress\)](#)
- [Mode shape recovery options \(fedem\\_modes\)](#)
- [Strain rosette recovery options \(fedem\\_gage\)](#)
- [Strain coat recovery options \(fedem\\_fpp\)](#)
- [Curve export options \(fedem\\_graphexp\)](#)

## C.1 Fedem GUI options

Command-line option	Description	Default value
<code>-checkRDB\interval</code>	Time [ms] between each RDB check/update during solve	500
<code>-console</code>	Enable console window	- (false)
<code>-debug</code>	Run in debug mode	- (false)
<code>-events</code>	Simulation event definition file	
<code>-exportAnimations</code>	Auto-export toggled animations to VTF on batch solve	- (false)
<code>-exportCurves</code>	Auto-export curves on batch solve. Specifies folder to export curve files into.	
<code>-f*</code>	Model file to open	<i>untitled.fmm</i>
<code>-help</code>	Display this help and exit	- (false)
<code>-licenseinfo</code>	Print out license information at startup	- (false)
<code>-logFile</code>	Write all Output List contents to log-file. Log-file name: <modelFilePrefix>.log	+ (true)
<code>-noAddOn</code>	Do not use licenses for add-on modules	- (false)
<code>-noFEData</code>	Load model file without FE-Models and FE-Results info. Use together with <code>-f</code>	- (false)
<code>-plotElements</code>	Enable plotting of element results	- (false)
<code>-plotNodes</code>	Enable plotting of nodal results	- (false)
<code>-prepareBatch</code>	Prepare for batch execution. This option can have the following arguments: all = all solvers reducer = reduction of all parts dynamics = dynamics solver stress = stress recovery for in parts modes = mode shape recovery in all parts strangage = strain gage recovery in all parts straincoat = strain coat recovery in all parts	
<code>-purgeOnSave</code>	Purge inactive mechanism objects on Save	- (false)
<code>-solve</code>	Start given solver(s) in batch mode. This option can have the following arguments: all = all solvers events = all solvers on all events reducer = reduction of all parts dynamics = dynamics solver stress = stress recovery in all parts modes = mode shape recovery in all parts strangage = strain gage recovery in all parts straincoat = strain coat recovery in all parts	

Command-line option	Description	Default value
<code>-timerange</code>	Time specification for batch stress recovery. Format: [startTime,stopTime[,timeInc]]. timeInc is optional.	
<code>-version</code>	Display program version and exit	<code>- (false)</code>

\* The `-f <modelfile>` option can also be specified without the `-f` flag if no other options are needed, i.e., the command `fedem mymodel.fmm` is equivalent to `fedem -f mymodel.fmm`.

## C.2 Reducer options (fedem\_reducer)

Command-line option	Description	Default value
<code>-autoMassScale</code>	Scale factor for auto-added masses	<code>1e-009</code>
<code>-autoStiffMethod</code>	Method for automatic stiffness computations in auto-added springs = 1: $k = \text{Min}(\text{diag}(K)) * 0.1 / \langle \text{tolFactorize} \rangle$ = 2: $k = \text{Mean}(\text{diag}(K)) * \langle \text{autoStiffScale} \rangle$ = 3: $k = \text{Max}(\text{diag}(K)) * \langle \text{autoStiffScale} \rangle$	3
<code>-autoStiffScale</code>	Scale factor for auto-added springs	100
<code>-Bmatfile</code>	Name of B-matrix file	
<code>-Bmatprecision</code>	Storage precision of the B-matrix on disk = 1: Single precision = 2: Double precision	2
<code>-Bramsize</code>	In-core size (MB) of displacement recovery matrix $\leq 0$ : Store full matrix in core	-1
<code>-bufsize_rigid</code>	Buffer size (in DP-words) per rigid element $\leq 0$ : Use conservative estimate computed internally	0
<code>-cachesize</code>	Cache size (KB) to be used by the SPR solver. Applies to the stiffness matrix only when lumped mass is used.	0
<code>-consolemsg</code>	Output error messages to console	- ( <i>false</i> )
<code>-cwd</code>	Change working directory	
<code>-datacheck</code>	Do data check only (exiting after data input)	- ( <i>false</i> )
<code>-debug</code>	Debug print switch	0
<code>-denseSolver</code>	Use LAPACK dense matrix equation solver	- ( <i>false</i> )
<code>-dispfile</code>	Name of displacement vector file	
<code>-drillingStiff</code>	Fictitious drilling DOF stiffness	<code>1e-006</code>
<code>-eigenshift</code>	Shift value for eigenvalue analysis (target frequency for generalized DOFs)	0
<code>-eigfile</code>	Name of eigenvector file	
<code>-extNodes</code>	List of external nodes to use in the reduction (in addition to the nodes specified in the FE data file)	
<code>-factorMass</code>	Factorize mass matrix in the eigensolver	- ( <i>false</i> )
<code>-fao</code>	Read additional options from this file	
<code>-fco</code>	Read calculation options from this file	

Command-line option	Description	Default value
<code>-fop</code>	Read output options from this file	
<code>-frsfile</code>	Name of results database file for mode shapes	
<code>-ftlout</code>	Name of part output file in FTL-format	
<code>-gravfile</code>	Name of gravity force vector file	
<code>-help</code>	Print out this help text	- ( <i>false</i> )
<code>-licenseinfo</code>	Print out license information at startup	- ( <i>false</i> )
<code>-licensepath</code>	License file directory	
<code>-linkId</code>	Part base-ID number	1
<code>-linkfile</code>	Name of FE data file (must be specified)	
<code>-loadfile</code>	Name of load vector file	
<code>-lumpedmass</code>	Use lumped element mass matrices	- ( <i>false</i> )
<code>-massfile</code>	Name of mass matrix file	
<code>-neval</code>	Number of eigenvalues/eigenvectors to compute	0
<code>-nevred</code>	Number of eigenvalues to compute for reduced system	12
<code>-ngen</code>	Number of generalized modes	0
<code>-nograv</code>	Skip gravity force calculation and mass matrix reduction	- ( <i>false</i> )
<code>-nomass</code>	Skip mass matrix reduction	- ( <i>false</i> )
<code>-printArray</code>	Additional debug print switch for certain arrays	0
<code>-rdbinc</code>	Increment number for the results data-base files	1
<code>-resfile</code>	Name of result output file	
<code>-samfile</code>	Name of SAM data file	
<code>-singularityHandler</code>	Option on how to treat singular matrices = 0: Abort on all occurring singularities = 1: Suppress true zero pivots, abort on reduced-to-zero pivots > 1: Suppress all occurring singularities of any kind	1
<code>-skylinesolver</code>	Use the skyline equation solver	- ( <i>false</i> )
<code>-stifffile</code>	Name of stiffness matrix file	
<code>-terminal</code>	File unit number for terminal output	6

Command-line option	Description	Default value
<code>-tolEigval</code>	Max acceptable relative error in eigen-values	<code>1e-008</code>
<code>-tolFactorize</code>	Equation solver singularity criterion (smaller values are less restrictive). The lowest value allowed is <code>1e-020</code> .	<code>1e-012</code>
<code>-tolWAVGM</code>	Geometric tolerance for WAVGM elements	<code>0.0001</code>
<code>-version</code>	Print out program version	<code>- (false)</code>

## C.3 Dynamics solver options (fedem\_solver)

<b>Command-line option</b>	<b>Description</b>	<b>Default value</b>
<code>-addBC_eigensolver</code>	Use additional BCs on eigensolver	- (false)
<code>-allAccVars</code>	Output all acceleration variables	- (false)
<code>-allAlgorVars</code>	Output all algorithm variables	- (false)
<code>-allBeamForces</code>	Output all beam sectional forces	- (false)
<code>-allCGVars</code>	Output all centre of gravity variables	- (false)
<code>-allControlVars</code>	Output all control variables	- (false)
<code>-allDampCoeff</code>	Output all damper coefficients	- (false)
<code>-allDamperVars</code>	Output all damper variables	- (false)
<code>-allDefVars</code>	Output all deflection variables	- (false)
<code>-allEnergyVars</code>	Output all energy quantities	- (false)
<code>-allEngineVars</code>	Output all engine values	- (false)
<code>-allForceVars</code>	Output all force variables	- (false)
<code>-allFrictionVars</code>	Output all friction variables	- (false)
<code>-allGages</code>	Output all strain gages	- (false)
<code>-allGenDOFVars</code>	Output all generalized DOF variables	- (false)
<code>-allHDVars</code>	Output all hydrodynamics variables	- (false)
<code>-allJointVars</code>	Output all joint variables	- (false)
<code>-allLengthVars</code>	Output all length variables	- (false)
<code>-allLoadVars</code>	Output all external load variables	- (false)
<code>-allPrimaryVars</code>	Output all primary variables	+ (true)
<code>-allRestartVars</code>	Output all variables needed for restart	- (false)
<code>-allSecondaryVars</code>	Output all secondary variables	- (false)
<code>-allSpringVars</code>	Output all spring variables	- (false)
<code>-allStiffVars</code>	Output all spring stiffnesses	- (false)
<code>-allSupelVars</code>	Output all superelement variables	- (false)
<code>-allSystemVars</code>	Output all system variables	- (false)
<code>-allTireVars</code>	Output all tire variables	- (false)
<code>-allTriadVars</code>	Output all triad variables	- (false)
<code>-allVelVars</code>	Output all velocity variables	- (false)
<code>-alphaNewmark</code>	Time integration parameters	0.1
<code>-alpha1</code>	Global mass-proportional damping factor	0.0

Command-line option	Description	Default value
<code>-alpha2</code>	Global stiffness-proportional damping factor	0.0
<code>-autoTimeStep</code>	Time stepping procedure = 0: Fixed time step size = 1: Automatically computed time step size	0
<code>-Bramsize</code>	In-core size (MB) of the displacement recovery matrix < 0: Use the same as in the reducer = 0: Store full matrix in core	-1
<code>-centripForceCorr</code>	Use centripetal force correction	- ( <i>false</i> )
<code>-consolemsg</code>	Output error messages to console	- ( <i>false</i> )
<code>-ctrlTol</code>	Control system tolerance parameters (1)=Absolute iteration tolerance (2)=Relative iteration tolerance (3)=Accuracy parameter (4)=Relative perturbation for numerical Jacobian computation	0.002 0.002 0.5 1e-005
<code>-ctrlfile</code>	Name of control system database file	<code>ctrl.frs</code>
<code>-curveFile</code>	Name of curve definition file	<code>response.bak.fmm</code>
<code>-curvePlotFile</code>	Name of curve export output file	
<code>-curvePlotPrec</code>	Output precision for exported curve data files = 0 : half precision (int*2) = 1 : single precision (real*4) = 2 : double precision (real*8)	0
<code>-curvePlotType</code>	Format of curve export output file = 0 : ASCII (separate file for each curve) = 1 : DAC, Windows (separate file for each curve) = 2 : DAC, UNIX (separate file for each curve) = 3 : RPC, Windows (all curves in one file) = 4 : RPC, UNIX (all curves in one file) = 5 : ASCII (all curves in one file)	0
<code>-cutbackFactor</code>	Time step reduction factor in cut-back	1
<code>-cutbackNegPiv</code>	Try cut-back when detecting negative pivots	100
<code>-cutbackSing</code>	Try cut-back when detecting singularities	- ( <i>false</i> )
<code>-cutbackSteps</code>	Number of cut-back steps	0
<code>-cwd</code>	Change working directory	
<code>-damped</code>	Solve the damped eigenproblem using LAPACK	- ( <i>false</i> )

<b>Command-line option</b>	<b>Description</b>	<b>Default value</b>
<code>-debug</code>	Debug print switch	0
<code>-delayBuffer</code>	Initial buffer size for delay elements	1000
<code>-densesolver</code>	Use LAPACK dense matrix equation solver	- (false)
<code>-diffraction</code>	Target number of diffraction panels	0
<code>-displacementfile</code>	Name of file with boundary displacements	
<code>-double1</code>	Save primary variables in double precision	+ (true)
<code>-double2</code>	Save secondary variables in double precision	- (false)
<code>-effModalMass</code>	Compute the effective mass for each mode	- (false)
<code>-eigenshift</code>	Shift value for vibration eigenvalue analysis (negative value captures zero frequency modes)	0
<code>-eiginc</code>	Time between each eigenvalue analysis	0
<code>-externalfuncfile</code>	Name of external function value file	
<code>-factorMass_eigen\solver</code>	Factor mass matrix in eigensolver	- (false)
<code>-fao</code>	Read additional options from this file	
<code>-fco</code>	Read calculation options from this file	
<code>-flushinc</code>	Time between each database file flush < 0.0: Do not flush results database (let the OS decide) = 0.0: Flush at each time step, no external buffers > 0.0: Flush at specified time interval, use external buffers	0
<code>-fop</code>	Read output options from this file	
<code>-frequency_domain</code>	Switch for frequency domain solution	- (false)
<code>-freqfile</code>	Name of frequency response database file	
<code>-frs1file</code>	Name of primary response database file	<i>th_p.frs</i>
<code>-frs2file</code>	Name of secondary response database file	<i>th_s.frs</i>
<code>-frs3file</code>	Name of stress- and gage recovery database files	
<code>-fsi2file</code>	Name of additional solver input file	
<code>-fsifile</code>	Name of solver input file	<i>fedem_solver.fsi</i>
<code>-help</code>	Print out this help text	- (false)
<code>-ignoreIC</code>	Ignore initial conditions from the fsi-file	- (false)

Command-line option	Description	Default value
<code>-initEqAD</code>	Initial equilibrium with aerodynamic loads	- ( <i>false</i> )
<code>-initEquilibrium</code>	Initial static equilibrium iterations	- ( <i>false</i> )
<code>-lancz1</code>	Use the LANCZ1 eigensolver	- ( <i>false</i> )
<code>-licenseinfo</code>	Print out license information at startup	- ( <i>false</i> )
<code>-licensepath</code>	License file directory	
<code>-limInitEquilStep</code>	Initial equilibrium step size limit	1
<code>-lineSearch</code>	Use line search in the nonlinear iterations	- ( <i>false</i> )
<code>-maxInc</code>	Maximum time increment	0.05
<code>-maxSeqNoUpdate</code>	Max number of sequential iterations without system matrix update	100
<code>-maxit</code>	Maximum number of iterations	15
<code>-minInc</code>	Minimum time increment	0.001
<code>-minit</code>	Minimum number of iterations	1
<code>-modesfile</code>	Name of primary modes database file	<code>ev_p.frs</code>
<code>-monitorIter</code>	Number of iterations to monitor before maxit	2
<code>-monitorWorst</code>	Number of DOFs to monitor on poor convergence	6
<code>-NewmarkFlag</code>	Newmark time integration option (=cba) a > 0: Compute inertia force from residual of previous increment in calculation of the right-hand-side vector b > 0: Use total solution increment in configuration update c = 1: Use HHT-alpha algorithm equivalent to FENRIS c = 2: Use generalized-alpha algorithm with interpolated internal forces	0
<code>-noBeamForces</code>	Suppress all beam sectional force output	- ( <i>false</i> )
<code>-nosolveropt</code>	Switch off equation system reordering	- ( <i>false</i> )
<code>-nrModes</code>	Switch between modal or direct frequency response solution, if 0 direct solution	0
<code>-numEigModes</code>	Number of eigenmodes to calculate	0
<code>-num_damp_energy_\skip</code>	Number of steps without calculation of energy from stiffness proportional damping	1
<code>-nunit</code>	Fixed number of iterations	0
<code>-nupdat</code>	Number of iterations with system matrix update	0

<b>Command-line option</b>	<b>Description</b>	<b>Default value</b>
<code>-pardiso</code>	Use the Pardiso sparse equation solver	- ( <i>false</i> )
<code>-pardisoIndef</code>	Use Pardiso, indefinite system matrices	- ( <i>false</i> )
<code>-partDeformation</code>	Output recovered part deformations (0=off, 1=local, 2=total, 3=local and total)	1
<code>-partVMStress</code>	Output recovered von Mises stresses on parts (0=off, 1=on)	1
<code>-plugin</code>	Plugin(s) for user-defined elements and functions	
<code>-printFunc</code>	Option for function output. 1: Print wave spectrum to result output file 2: Print function evaluation to separate file	0
<code>-printTriad</code>	Print some triads to result output file	
<code>-printinc</code>	Time between each print to result output file	0
<code>-quasiStatic</code>	Do a quasi-static simulation to this time	0
<code>-rampData</code>	Ramp-up function parameters (1)=max speed during ramp-up stage (2)=total length (in time) of ramp-up stage (3)=time after ramp-up before new load	1.0 2.0 0.0
<code>-rampGravity</code>	Ramp up gravity forces also	- ( <i>false</i> )
<code>-rampSteps</code>	Number of increments in ramp-up stage	0
<code>-rdbinc</code>	Increment number for the results database files	1
<code>-rdblength</code>	Maximum time length of results database files	0
<code>-recovery</code>	Recovery option (0=off, 1=stress, 2=gage, 3=both)	0
<code>-resfile</code>	Name of result output file	<i>fedem_solver.res</i>
<code>-restartfile</code>	Response database file(s) to restart from	
<code>-restarttime</code>	Physical time for restart < 0: No restart, but regular simulation	-1
<code>-rpcFile</code>	Get number of repeats, averages, and points per frame and group, from this RPC-file	
<code>-sample_freq</code>	Define sampling frequency	100.0
<code>-saveinc2</code>	Time between each save of secondary variables	0
<code>-saveinc3</code>	Time between each save for external recovery	0

Command-line option	Description	Default value
<code>-saveinc4</code>	Time between each save of control system data	0
<code>-savestart</code>	Time for first save to response database	0
<code>-scaleToKG</code>	Scaling factor to SI mass unit [kg]	1
<code>-scaleToM</code>	Scaling factor to SI length unit [m]	1
<code>-scaleToS</code>	Scaling factor to SI time unit [s]	1
<code>-skylinesolver</code>	Use skyline solver	- ( <i>false</i> )
<code>-stiffDamp\Filtering</code>	Rigid body filtering of stiffness-proportional damping = 0: Deactivated = 1: Use first-order approximation of deformational velocity = 2: Use second-order approximation of deformational velocity (based on Newmark update formula)	1
<code>-stopGlbDmp</code>	Stop time for global structural damping factors	-1.0
<code>-stopOnDivergence</code>	Number of warnings on possible divergence before the dynamics simulation is aborted (0 = no limit)	0
<code>-stressStiff\DivergSkip</code>	Number of iterations without stress stiffening on cut-back with same step size	0
<code>-stressStiffDyn</code>	Use geometric stiffness for dynamics	- ( <i>false</i> )
<code>-stressStiffEig</code>	Use geometric stiffness for eigenvalue analysis	- ( <i>false</i> )
<code>-stressStiffEqu</code>	Use geometric stiffness for statics	- ( <i>false</i> )
<code>-stressStiff\UpdateSkip</code>	Number of iterations without updating stress stiffening (always updated in predictor step)	0
<code>-targetFrequency\Rigid</code>	Target frequency for auto-stiffness calculation	10000
<code>-terminal</code>	File unit number for terminal output	6
<code>-timeEnd</code>	Stop time	0
<code>-timeInc</code>	Initial time increment	0
<code>-timeStart</code>	Start time	0
<code>-tolAccGen</code>	Max generalized acceleration tolerance	0 *
<code>-tolAccNorm</code>	Acceleration vector tolerance	0 *
<code>-tolAccRot</code>	Max angular acceleration tolerance	0 *
<code>-tolAccTra</code>	Max acceleration tolerance	0 *

<b>Command-line option</b>	<b>Description</b>	<b>Default value</b>
<code>-tolDispGen</code>	Max generalized DOF tolerance	0 *
<code>-tolDispNorm</code>	Displacement vector tolerance	0 *
<code>-tolDistRot</code>	Max rotation tolerance	0 *
<code>-tolDispTra</code>	Max displacement tolerance	0 *
<code>-tolEigval</code>	Max acceptable relative error in eigen-values	1e-008
<code>-tolEigvector</code>	Orthogonality limit for the eigenvectors	1e-008
<code>-tolEnerMax</code>	Max energy in a single DOF tolerance	0 *
<code>-tolEnerSum</code>	Energy norm convergence tolerance	0 *
<code>-tolFactorize</code>	Linear solver singularity criteria (1)=time domain simulation (2)=initial static equilibrium and eigen-value analysis (3)=control system integration (smaller values less restrictive)	1e-012 1e-009 1e-012
<code>-tolInitEquil</code>	Convergence tolerance for initial equilibrium iterations	0.001
<code>-tolResGen</code>	Max residual generalized DOF force tolerance	0 *
<code>-tolResNorm</code>	Residual force vector tolerance	0 *
<code>-tolResRot</code>	Max residual torque tolerance	0 *
<code>-tolResTra</code>	Max residual force tolerance	0 *
<code>-tolUpdateFactor</code>	Convergence criterion for continuing matrix updates	0 *
<code>-tolVelGen</code>	Max generalized velocity tolerance	0 *
<code>-tolVelNorm</code>	Velocity vector convergence tolerance	0 *
<code>-tolVelRot</code>	Max angular velocity tolerance	0 *
<code>-tolVelTra</code>	Max velocity tolerance	0 *
<code>-version</code>	Print out program version	- (false)
<code>-VTFfile</code>	Name of VTF output file	
<code>-windowSize</code>	Defines the window size in samples, for frequency response analysis	0
<code>-yamlFile</code>	YAML file prefix for system mode shape export	

- \* For all the convergence tolerance options, its value is interpreted as follows:
  - = 0 : This tolerance is ignored
  - > 0 : This tolerance is in a set of tests where all must be satisfied
  - < 0 : This tolerance is in a set of tests where only one must be satisfied  
(using the absolute value as the actual tolerance value)

## C.4 Stress recovery options (fedem\_stress)

Command-line option	Description	Default value
<code>-Bmatfile</code>	Name of B-matrix file	
<code>-Bramsize</code>	In-core size (MB) of displacement recovery matrix < 0: Use the same as in the reducer = 0: Store full matrix in core	-1
<code>-consolemsg</code>	Output error messages to console	- (false)
<code>-cwd</code>	Change working directory	
<code>-debug</code>	Debug print switch	0
<code>-deformation</code>	Save deformations to results database	- (false)
<code>-double</code>	Save all results in double precision	- (false)
<code>-dumpDefNas</code>	Save deformations to Nastran bulk data files	- (false)
<code>-dispfile</code>	Name of gravitation displacement file	
<code>-eigfile</code>	Name of eigenvector file	
<code>-fao</code>	Read additional options from this file	
<code>-fco</code>	Read calculation options from this file	
<code>-fop</code>	Read output options from this file	
<code>-frsfile</code>	Name of solver results database file	
<code>-fsifile</code>	Name of solver input file	<code>fedem_solver.fsi</code>
<code>-group</code>	List of element groups to do calculations for	
<code>-help</code>	Print out this help text	- (false)
<code>-licenseinfo</code>	Print out license information at startup	- (false)
<code>-licensepath</code>	License file directory	
<code>-linkId</code>	Part base-ID number	0
<code>-linkfile</code>	Name of FE data file	
<code>-maxPStrain</code>	Save max principal strain to results database	- (false)
<code>-maxPStress</code>	Save max principal stress to results database	- (false)
<code>-maxSStrain</code>	Save max shear strain to results database	- (false)
<code>-maxSStress</code>	Save max shear stress to results database	- (false)
<code>-minPStrain</code>	Save min principal strain to results database	- (false)

<b>Command-line option</b>	<b>Description</b>	<b>Default value</b>
<code>-minPStress</code>	Save min principal stress to results database	- ( <i>false</i> )
<code>-nodalForces</code>	Compute and print nodal forces	- ( <i>false</i> )
<code>-rdbfile</code>	Name of stress results database file	
<code>-rdbinc</code>	Increment number for the results database file	1
<code>-resfile</code>	Name of result output file	
<code>-resStressFile</code>	Name of residual stress input file	
<code>-resStressSet</code>	Name of residual stress set	
<code>-samfile</code>	Name of SAM data file	
<code>-SR</code>	Save stress resultants to results database	- ( <i>false</i> )
<code>-statm</code>	Start time	0
<code>-stotm</code>	Stop time	1
<code>-strain</code>	Save strain tensors to results database	- ( <i>false</i> )
<code>-stress</code>	Save stress tensors to results database	- ( <i>false</i> )
<code>-terminal</code>	File unit number for terminal output	6
<code>-tinc</code>	Time increment (= 0.0: process all time steps)	0.1
<code>-version</code>	Print out program version	- ( <i>false</i> )
<code>-vmStrain</code>	Save von Mises strain to results database	- ( <i>false</i> )
<code>-vmStress</code>	Save von Mises stress to results database	- ( <i>false</i> )
<code>-VTFavgelm</code>	Write averaged element results to VTF-file	+ ( <i>true</i> )
<code>-VTFdscale</code>	Deformation scaling factor for VTF output	1
<code>-VTFfile</code>	Name of VTF output file	
<code>-VTFinit</code>	Write initial state to VTF-file	- ( <i>false</i> )
<code>-VTFoffset</code>	VTF result block id offset	0
<code>-VTFparts</code>	Number of parts in VTF-file	0
<code>-write_nodes</code>	Save deformations as nodal data	+ ( <i>true</i> )
<code>-write_vector</code>	Save deformations as vector data	- ( <i>false</i> )

## C.5 Mode shape recovery options (fedem\_modes)

Command-line option	Description	Default value
<code>-Bmatfile</code>	Name of B-matrix file	
<code>-Bramsize</code>	In-core size (MB) of displacement recovery matrix < 0: Use the same as in the reducer = 0: Store full matrix in core	-1
<code>-consolemsg</code>	Output error messages to console	- (false)
<code>-cwd</code>	Change working directory	
<code>-damped</code>	Complex modes are calculated	- (false)
<code>-debug</code>	Debug print switch	0
<code>-dispfile</code>	Name of gravitation displacement file	
<code>-double</code>	Save all results in double precision	- (false)
<code>-eigfile</code>	Name of eigenvector file	
<code>-energy_density</code>	Save scaled strain energy density	- (false)
<code>-fao</code>	Read additional options from this file	
<code>-fco</code>	Read calculation options from this file	
<code>-fop</code>	Read output options from this file	
<code>-frsfile</code>	Name of solver results database file	
<code>-fsifile</code>	Name of solver input file	<code>fedem_solver.fsi</code>
<code>-help</code>	Print out this help text	- (false)
<code>-licenseinfo</code>	Print out license information at startup	- (false)
<code>-licensepath</code>	License file directory.	
<code>-linkId</code>	Part base-ID number	0
<code>-linkfile</code>	Name of FE data file	
<code>-rdbfile</code>	Name of modes results database file	
<code>-rdbinc</code>	Increment number for the results database file	1
<code>-recover_modes</code>	List of mode numbers to expand	
<code>-resfile</code>	Name of result output file	
<code>-samfile</code>	Name of SAM data file	
<code>-terminal</code>	File unit number for terminal output	6
<code>-version</code>	Print out program version	- (false)
<code>-VTFdscale</code>	Deformation scaling factor for VTF output	1
<code>-VTExpress</code>	Write express VTF-files (one file per mode)	- (false)

Command-line option	Description	Default value
<code>-VTFfile</code>	Name of VTF output file	
<code>-VTFoffset</code>	VTF result block id offset	0
<code>-VTFparts</code>	Number of parts in VTF-file	0
<code>-write_nodes</code>	Save results as nodal data	- ( <i>false</i> )
<code>-write_vector</code>	Save results as vector data	+ ( <i>true</i> )

## C.6 Strain rosette recovery options (fedem\_gage)

Command-line option	Description	Default value
<code>-binSize</code>	Bin size for stress cycle counting [MPa]	10
<code>-Bmatfile</code>	Name of B-matrix file	
<code>-Bramsize</code>	In-core size (MB) of displacement recovery matrix < 0: Use the same as in the reducer = 0: Store full matrix in core	-1
<code>-consolemsg</code>	Output error messages to console	- (false)
<code>-cwd</code>	Change working directory	
<code>-dac_sampleinc</code>	Sampling increment for dac output files	0.001
<code>-debug</code>	Debug print switch	0
<code>-deformation</code>	Save nodal deformations to results database	- (false)
<code>-displfile</code>	Name of gravitation displacement file	
<code>-eigfile</code>	Name of eigenvector file	
<code>-fao</code>	Read additional options from this file	
<code>-fatigue</code>	Perform damage calculation on the gage stresses	0
<code>-fco</code>	Read calculation options from this file	
<code>-flushinc</code>	Time between each database file flush < 0.0: Do not flush results database (let the OS decide) = 0.0: Flush at each time step, no external buffers > 0.0: Flush at specified time interval, use external buffers	-1
<code>-fop</code>	Read output options from this file	
<code>-frsfile</code>	Name of solver results database file	
<code>-fsifile</code>	Name of solver input file	<code>fedem_solver.fsi</code>
<code>-gate</code>	Stress gate value for the damage calculation [MPa]	25
<code>-help</code>	Print out this help text	- (false)
<code>-licenseinfo</code>	Print out license information at startup	- (false)
<code>-licensepath</code>	License file directory	
<code>-linkId</code>	Part base-ID number	0
<code>-linkfile</code>	Name of FE data file	
<code>-littleEndian</code>	Use Little Endian formatting of DAC files	- (false)*

Command-line option	Description	Default value
<code>-loga1</code>	Parameter log(a1) of the S-N curve	15.117
<code>-loga2</code>	Parameter log(a2) of the S-N curve	17.146
<code>-m1</code>	Parameter m1 of the S-N curve	4
<code>-nullify_start_\rosettestrains</code>	Set start strains to zero for the rosettes	- ( <i>false</i> )
<code>-rdbfile</code>	Name of strain gage results database file	
<code>-rdbinc</code>	Increment number for the results database file	1
<code>-resfile</code>	Name of result output file	
<code>-rosfile</code>	Name of strain rosette input file	
<code>-samfile</code>	Name of SAM data file	
<code>-statm</code>	Start time	0
<code>-stotm</code>	Stop time	1
<code>-stressToMPaScale</code>	Scale factor scaling stresses to MPa	1e-006
<code>-terminal</code>	File unit number for terminal output	6
<code>-tinc</code>	Time increment (= 0.0: process all time steps)	0
<code>-version</code>	Print out program version	- ( <i>false</i> )
<code>-writeAsciiFiles</code>	Write rosette results to ASCII files	- ( <i>false</i> )

\* This is the default value on UNIX platforms. On Windows, the default is + (*true*). Thus, the default formatting of the DAC files will be suitable for the platform the recovery is run on.

## C.7 Strain coat recovery options (fedem\_fpp)

Command-line option	Description	Default value
<code>-angleBins</code>	Number of bins in search for most popular angle	541
<code>-biAxialGate</code>	Gate value for the biaxiality calculation	10
<code>-blockSize</code>	Max number of elements processed together	2000
<code>-Bmatfile</code>	Name of B-matrix file	
<code>-Bramsize</code>	In-core size (MB) of displacement recovery matrix < 0: Use the same as in the reducer = 0: Store full matrix in core	-1
<code>-BufSizeInc</code>	Buffer increment size	20
<code>-consolemsg</code>	Output error messages to console	- (false)
<code>-cwd</code>	Change working directory	
<code>-debug</code>	Debug print switch	0
<code>-dispfile</code>	Name of gravitation displacement file	
<code>-double</code>	Save results in double precision	- (false)
<code>-eigfile</code>	Name of eigenvector file	
<code>-fao</code>	Read additional options from this file	
<code>-fco</code>	Read calculation options from this file	
<code>-fop</code>	Read output options from this file	
<code>-fppfile</code>	Name of fpp output file	
<code>-frsfile</code>	Name of solver results database file	
<code>-fsifile</code>	Name of solver input file	<code>fedem_solver.fsi</code>
<code>-group</code>	List of element groups to do calculations for	
<code>-help</code>	Print out this help text	- (false)
<code>-HistDataType</code>	Histogram data type = 0: None = 1: Signed abs max stress = 2: Signed abs max strain	0
<code>-HistXBins</code>	Histogram number of X-bins	64
<code>-HistXMax</code>	Histogram max X-value	100
<code>-HistXMin</code>	Histogram min X-value	-100
<code>-HistYBins</code>	Histogram number of Y-bins	64
<code>-HistYMax</code>	Histogram max Y-value	100

Command-line option	Description	Default value
<code>-HistYMin</code>	Histogram min Y-value	-100
<code>-licenseinfo</code>	Print out license information at startup	- ( <i>false</i> )
<code>-licensepath</code>	License file directory	
<code>-linkId</code>	Part base-ID number	0
<code>-linkfile</code>	Name of FE data file	
<code>-PVXGate</code>	Gate value for the Peak Valley extraction (MPa or microns depending on HistData-Type)	10
<code>-rdbfile</code>	Name of strain coat results database file	
<code>-rdbinc</code>	Increment number for the results database file	1
<code>-resfile</code>	Name of result output file	
<code>-resStressFile</code>	Name of residual stress input file	
<code>-resStressSet</code>	Name of residual stress set	
<code>-samfile</code>	Name of SAM data file	
<code>-SNfile</code>	Name of SN-curve definition file	
<code>-statm</code>	Start time	0
<code>-stotm</code>	Stop time	1
<code>-stressToMPaScale</code>	Stress conversion factor to MPa	$1e-06$
<code>-surcface</code>	Surface selection option = 0: All element surfaces = 1: Bottom shell surfaces only = 2: Middle shell surfaces only = 3: Top shell surfaces only	0
<code>-terminal</code>	File unit number for terminal output	6
<code>-tinc</code>	Time increment (= 0.0: process all time steps)	0
<code>-version</code>	Print out program version	- ( <i>false</i> )

## C.8 Curve export options (fedem\_graphexp)

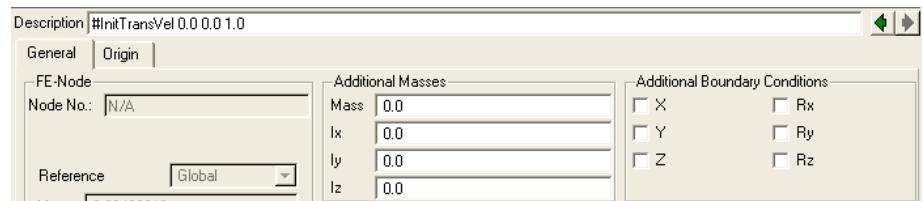
Command-line option	Description	Default value
<code>-curvePlotFile</code>	Name of curve export output file	<code>response.rsp</code>
<code>-curvePlotPrec</code>	Output precision for curve data files = 0 : half precision (int*2) = 1 : single precision (real*4) = 2 : double precision (real*8)	0*
<code>-curvePlotType</code>	Format of curve export output file = 0 : ASCII (separate file for each curve) = 1 : DAC, Windows (separate file for each curve) = 2 : DAC, UNIX (separate file for each curve) = 3 : RPC, Windows (all curves in one file) = 4 : RPC, UNIX (all curves in one file) = 5 : ASCII (all curves in one file)	3
<code>-cwd</code>	Change working directory	
<code>-fao</code>	Read additional options from this file	
<code>-fco</code>	Read calculation options from this file	
<code>-fop</code>	Read output options from this file	
<code>-frsFile</code>	List of results database files	
<code>-help</code>	Display this help and exit	- ( <i>false</i> )
<code>-licenseinfo</code>	Print out license information at startup	- ( <i>false</i> )
<code>-licensepath</code>	License file directory	
<code>-modelFile</code>	Name of model file with curve definitions	
<code>-rpcFile</code>	Get number of repeats, averages, and points per frame and group, from this RPC-file	
<code>-version</code>	Display program version and exit	- ( <i>false</i> )

\* The `-curvePlotPrec` option is effective for multi-column ASCII and RPC files only. The default value 0 (half precision) is applicable to RPC files only. For ASCII files, the default value is 1 (single precision). DAC files and single-column ASCII files are always written in single precision. This footnote also applies to the `-curvePlotPrec` option of the Dynamics Solver (see [Section C.3, "Dynamics solver options \(fedem\\_solver\)"](#)).



# Appendix D Beta feature documentation

Some new features in Fedem are still in a state of Beta testing. These features are typically available through commands or options entered in the description field of an entity's property panel. This means to enter the special character "#" followed by a keyword and possibly some values into the description field along with the user description, like in the example shown below. See also [Section 2.4.5, "Property Editor"](#).



Due to the Beta nature of the features, they should be used with some care. The present way of accessing these features through the options or commands in the description field is subject to change, or no longer being supported, in future releases. However, most of the features listed here will be supported in a more "permanent" fashion in future releases.

Sections in this appendix address the following topics:

- [Joints](#)
- [Parts](#)
- [Beams](#)
- [Beam properties](#)
- [Springs](#)
- [Frictions](#)
- [Additional masses](#)
- [External loads](#)
- [Sensors](#)
- [Wave functions](#)
- [Result output control](#)
- [Undocumented features](#)

## D.1 Joints

### D.1.1 Universal Joint

The universal joint is obtained from the Ball Joint when the command

```
#UniversalJoint
```

is entered in the description field of the selected Ball Joint. The initial orientation of the joint cross is assumed to be the Y and Z-axis of the Ball Joint master triad. The Z-axis of the cross is connected to the master part whereas the Y-axis is connected to the slave part. The user must thus reorient the master triad appropriately for the desired joint configuration. Attention must also be paid to the selection of master versus slave part.

### D.1.2 Constant Velocity Joint

A constant velocity joint is obtained from a Ball Joint when the command

```
#CVJoint [#RZ <float>] [#RY <float>]
```

is entered in the description field of the selected Ball Joint.

The X-axis of the master triad should be oriented along the rotation axis of the master shaft (part). The direction of the slave shaft rotation axis, the X-axis, is determined by one, or both, of the optional commands

```
#RZ <float> #RY <float>
```

#RZ gives direction of slave X-axis relative to master X-axis by a rotation about the Z-axis of the master triad. #RY obtains the X-axis from the master X-axis by a rotation about the Y-axis of the master triad. Care should be exercised when applying both rotations since the angles are Euler-Z-Y rotations. It is recommended that the master triad is oriented such that one only needs to use one of the command #RZ and #RY.



**NOTE:** *The universal joint and constant velocity joint have both only two independent joint DOFs (the Y- and Z-rotations). Specifying a spring stiffness and/or damping for the third DOF (X-rotation) for these joint types is therefore meaningless, although that is possible in the Property Editor panel. Any spring/damper properties specified on this DOF will be silently ignored. The same is true for initial conditions.*

### D.1.3 Rigid Joint

The DOFs of the Rigid Joint can be released individually by entering one or more of the commands

```
#FreeX #FreeY #FreeZ #FreeRX #FreeRY #FreeRZ
```

in the description field of the selected Rigid Joint. The rotational parameterization is rotation axis components. All DOFs are referred to master triad coordinate system.

### D.1.4 Axial Joint

An Axial Joint is obtained from the Free Joint when the command

```
#Axial
```

is entered in the description field of the selected Free Joint. This is a single-DOF joint that works like an Axial Spring and/or Damper, except that the length between the two triads here is used as a joint DOF. The **Tx** tab of the joint property panel (see [Section 5.6.2, "Joint properties"](#)) is used to control the behavior of the length DOF. All the other DOF tabs of the property panel are not used when `#Axial` is specified.

The advantage of using an Axial Joint opposed to a combined Axial Spring and Damper is that you are able to prescribe the length between the two triads directly, without the need for stiff springs and associated damping, which might render the model numerically unstable. It also reduces the total number of DOFs in the mechanism compared with an Axial Spring/Damper (reduced by 5 for each Axial Joint). More details on the formulation of the Axial Joint may be found in the Fedem 7.6 Theory Guide, [Section 6.2.7, "Axial Joint"](#).

### D.1.5 Free Joint

By default, the translational- and rotational DOFs are coupled in a Free Joint. It is possible to remove this coupling by entering the following description field command

```
#noRotTransCoupling
```

The Free Joint will then not transfer any bending moments due to eccentric master and slave triad locations. It will therefore behave more like a Ball Joint, if the translational DOFs are fixed or have been assigned a high stiffness.

An alternative Free Joint formulation is available through the specification of the following description field command

*#GlobalSpring*

The default master-slave joint formulation is then not used (which for the Free Joint only is a transformation of the free DOFs from the slave Triad to the joint variables, and no constraining). With the alternative formulation the added spring (and damper) properties are applied directly in the global coordinate directions between the two triads instead, and no change in free variables is performed. There are no eccentricity contributions either when the force between the two triads is not attacking along their common axis. Consequently, this free joint is somewhat equivalent to a BUSH element on the system level (see the Fedem 7.6 Theory Guide, Section E.10, "BUSH").

It is also possible to specify coupling stiffnesses in a free joint when using the *#GlobalSpring* command. This is done by appending the following to the description:

*#K<i><j> <kij>*

where *<kij>* is the constant coupling stiffness between the local DOFs *<i>* and *<j>* in the free joint. It is not possible to specify non-constant coupling stiffness values.

### D.1.6 Prismatic Joint and Cylindric Joint

Both Prismatic and Cylindric Joint can take the command

*#Extended*

in the description field. Entering this command allows the follower to travel beyond the first and last triad of the track. When the follower is past the beginning of the track, the first two triads receive the contact force in a statically consistent manner. When the follower is past the last triad the last two triads receive the contact force.

By default, the contact force is interpolated linearly between the two triads on both side of the follower, based on their relative location. However, a cubic interpolation using the four closest master triads at any time can be used instead, by specifying the description field command

*#Cubic*

The default rotational parameterization of the Cylindric Joint is Euler-Z-Y-X rotations, but this can be changed to components of the rotation axis vector by entering the command

*#RotAxisParam*

in the description field of the selected Cylindric Joint. This command can also be applied to the Cam Joint when the alternative master-slave based formulation is used (see [Section D.1.7, "Cam Joint"](#) below).

### D.1.7 Cam Joint

The default cam formulation uses non-linear springs to model contact between the follower (slave triad) and the master triads along the cam curve. This formulation has independent DOFs at the follower and all master triads, and is able to handle large separation between cam and follower.

An alternative formulation, in which the motion of the follower is expressed in a curvilinear coordinate system surrounding the cam surface and the slave triad is treated more like a true slave of the master triads, is obtained by entering the following command in the description field:

*#MasterSlaveCam*

With this formulation one can also enter one, or both, of the commands:

*#FixX #FixY*

The joint DOFs in these directions are then eliminated (along with the joint springs and dampers), and the follower is restricted to follow the cam in that direction. Using these options can give a numerically more stable formulation when no separation between the cam and follower is expected (or possible). The master-slave cam formulation is only able to handle small separation (relative to the cam-segment radius) between follower and cam surface. A cam surface with corners can also cause problems with this alternative formulation.

## D.2 Parts

### D.2.1 Geometric stiffness

The **Geometric stiffness contribution** toggle in the *Integration tab* of the Dynamics Solver Setup dialog box (see [Section 8.5.2, "Dynamics Solver \(Advanced Mode\)"](#)) applies normally to all flexible parts in the model during the dynamics simulation. It is possible to override this setting for a specific part using the following description field commands:

- #*DynStressStiffening* – Enables geometric stiffness for this part
- #*NoDynStressStiffening* – Disables geometric stiffness for this part

The commands affect the dynamics simulation only. They have no effect during the initial equilibrium and eigenmode analyses. They have no effect for generic parts with automatic stiffness calculation toggled on.

### D.2.2 Component modes

The number of *Component modes* that is entered in the Property Editor panel for a selected part, defines how many component mode shapes that should be computed during the reduction of that part, see "[Reduction Options tab](#)" in [Section 5.1.5](#). Thus, whenever you change this number, the part might need to be reduced again. The default is to use all the computed component mode shapes as DOFs for the part during the dynamics simulation. However, it is possible to use only a subset of the computed modes, by specifying the following description field commands for the part:

- #*InclModes* <m1> <m2> ... – Use only these component modes
- #*ExclModes* <m1> <m2> ... – Use all but these component modes

The structural damping parameters entered in the Property Editor panel for a part, see "[Part tab](#)" in [Section 5.1.5](#), are by default applied to the whole stiffness- and mass matrix of the part. However, it is possible to specify individual Rayleigh damping factors for each of the component modes through the following description field commands:

- #*Alpha1* <c1> <c2> ... – Mass-proportional damping factors for the component modes
- #*Alpha2* <c1> <c2> ... – Stiffness-proportional damping factors for the component modes

Thus,  $<c1>$  is the damping factor applied to the first component mode used,  $<c2>$  is the factor for the second mode, etc. If you specify fewer such individual damping factors than the number of component modes being used, the last value entered is used for all the remaining modes.

### D.2.3 Initial velocities

A uniform initial translational velocity that applies to the entire model may be specified in the Model Preferences dialog box (see [Section 4.9, "Model preferences"](#)). An alternative is to specify initial conditions to individual parts, through the following description field command:

```
#InitTransVel <ux> <uy> <uz>
```

The values  $<ux>$ ,  $<uy>$  and  $<uz>$  are then applied to all triads attached to that part, including the automatically generated center of gravity triad for generic parts. This is overridden by any non-zero initial velocity specified in the Triad property panel though. For generic parts, the center of gravity triad can also be assigned an initial rotational velocity through a similar description field command

```
#InitRotVel <rx> <ry> <rz>
```

### D.2.4 Stress recovery on element groups during dynamics simulation

Doing stress recovery on entire FE parts during the dynamics simulation, as described in [Section 8.5.6, "Stress and strain recovery during time integration"](#), is time and memory consuming. Often it is sufficient to limit this calculation to a sub-set of the FE part, which will be more efficient.

Use one of the following description-field commands for stress recovery during the dynamics simulation on specified element groups in a FE part:

```
#recover-stress gid
#recover-stress <gid1,gid2,...>
#recover-stress <PMAT id1,PTHICK id2,...>
```

The first two commands will calculate internal displacements and von Mises stresses for all elements in one or several element groups ( $gid$  and  $gid<i>$  are user IDs of the element groups to do recovery on). The third command can be used when recovery on implicit element groups is wanted, defined via the specified element properties (see [Section 5.2, "Element groups"](#) for more on implicit element groups). Here,  $id<i>$  represents the IDs of the element group property .

## D.3 Beams

### D.3.1 Structural damping

The stiffness-proportional damping coefficient ( $\alpha_2$ ) for a Beam element may be defined via a Function, through the command

```
#StructDmpEngine <id>
```

where  $<id>$  is the base ID of the Function defining the damping factor.

The same command can also be applied on FE and Generic Parts.



**NOTE:** The base ID of a mechanism object is normally not visible in the Model Manager panel. To see them, you have to launch Fedem in debug mode (using command-line option `-debug`). Then the base ID appears in curly braces {} in the Objects list. You can also use the Object Browser dialog box, see [Section 2.4, "Object Browser"](#).

### D.3.2 Time-dependent stiffness scaling

The stiffness matrix of a Beam element can be adjusted using a time- or state-dependent scale function through the description field command:

```
#StiffScaleEngine <id>
```

The  $<id>$  is the base ID of the desired scaling function.

The same command can also be applied on Generic Parts. However, it is not recommended to use it on FE Parts if you also are going to perform stress recovery on that part. This is because the displacement recovery operator, calculated during the FE Part reduction, does not account for the stiffness scaling factor such that the calculation of internal deformation and stresses will be incorrect.

### D.3.3 Beam diagram curve domain adjustment

When plotting results along beams (see [Section 9.3, "Beam diagrams"](#)), all beam elements along a string from a selected start Triad or Beam are considered, until then end of the beam string or a joint between several beam elements is detected. To create a beam diagram that stops at any beam element before the end or a joint is reached, the following description field command is entered for the desired end element:

```
#Stop
```

A beam diagram curve can normally be started from a “natural” end of a beam string as described above. However, it is possible to start a curve from any intermediate beam element, if the following command is entered for the desired element:

```
#Start
```

## D.4 Beam properties

For beam cross section objects with Hydrodynamics enabled, one may specify a *Slamming* load coefficient through the following command:

```
#Cs <Cs> <Tcs>
```

Where  $<Cs>$  is the actual slamming coefficient and  $<Tcs>$  is the duration time of the slamming load.

## D.5 Springs

### D.5.1 Sign-dependent stiffness scaling

Spring stiffness for both axial and joint springs can be adjusted using scale functions. Different adjustment for positive and negative deflection of a spring is often desirable when adjusting the stiffness of, for instance, hydraulic cylinders. To achieve this, use the description field commands:

```
#PosStiffScaleEngine <id>
```

```
#NegStiffScaleEngine <id>
```

The  $<id>$  is the function’s base ID. These commands set the scale function only for the respective deflection state. If used, they will override the scale function selected through the Spring property panel, if any.



**TIP:** You can access the description field of a joint spring by double-clicking the desired spring entry in the Topology view of the actual joint.

### D.5.2 Plastic springs

For springs using a nonlinear force-deflection curve as stiffness characteristics, the following command may be used to define a cyclic-plastic behavior:

D

---

```
#Cyclic [<flag>]
```

where *<flag>* is an optional integer value to specify how the spring behaves during unloading.

- flag = 1 : Unloading occurs with initial tangent stiffness (for positive deflections only). This is the default if not flag value is given.
- flag = 2 : Reserved for future used.
- flag = 3 : Unloading occurs using the secant stiffness, for both positive and negative deflections.

### D.5.3 Spring failure

When a joint spring has been assigned failure criteria through an Advanced spring characteristics object (see [Section 5.9.6, "Advanced spring characteristics"](#)), it is often desirable when that spring fails (i.e., the spring force and stiffness both drop to zero for any deflection level), that all the other springs in the same joint having failure criteria assigned should fail in the same instant such that the entire joint is decoupled. This is accomplished by specifying the following description field command for the Advanced spring characteristics object that the spring refers to:

```
#FailAll
```

### D.5.4 Stiffness proportional damping

Axial springs can be assigned a stiffness-proportional damping coefficient by specifying the following description field command:

```
#Rayleigh <alpha2>
```

where *<alpha2>* is the stiffness-proportional damping coefficient.

### D.5.5 Pulley element

An axial spring can be assigned one (or more, up to 8) extra triads to simulate a pulley systems. This is done by specifying the following description field command for the Axial spring object:

```
#addTriads <id1> <id2> ...
```

where *<id1> <id2> ...* are the base id of the extra triads.

## D.6 Frictions

An alternative friction formulation is available by specifying the following command in the description field of a friction object:

```
#Kstick <k>
```

where *<k>* is the stiffness used to enforce no movement in the friction DOF when in stick condition. This formulation is based on the use of a spring with varying yield criterion (see [Section 5.9.6, "Advanced spring characteristics"](#)). That is, the Max Yield Force is taken as the maximum occurring friction force before the friction DOF is slipping (see the Fedem 7.6 Theory Guide, *Section 6.5.4 "Total friction"*). The equivalent load for the friction calculation may be specified explicitly instead of relying on the formulas defined in the Fedem 7.6 Theory Guide, *Section 6.5 "Joint friction"*. This is done using the description field command

```
#FrictionForceEngine <id>
```

where *<id>* is the base ID of a Function that defines the equivalent load.

## D.7 Additional masses

The additional masses are scaled using a Function when the command

```
#MassScaleEngine <id>
```

is entered in the description field for the selected Triad. The *<id>* is the function's base ID. The additional rotational inertias for the selected triad are also scaled using the same function.

The additional masses may be applied in a specified local triad direction only, by specifying one of the following commands in the Triad description field (the commands only affects the translational mass):

```
#MassDir <cx> <cy> <cz>
```

```
#MassX #MassY #MassZ
```

where *<cx>*, *<cy>* and *<cz>* defines the local triad direction the additional mass should be applied in. `#MassX` is equivalent to `#MassDir 1 0 0`, `#MassY` is equivalent to `#MassDir 0 1 0`, and `#MassZ` is equivalent to `#MassDir 0 0 1`.

An additional mass is regarded as a *virtual added mass* when

```
#AddedMass <mx> <my> <mz>
```

is entered in the description field of the Triad description field. The additional mass then contributes to the mass matrix and inertia forces only, and not to the gravitational force vector. The added mass terms then equal { $<M> * <mx>$ ,  $<M> * <my>$ ,  $<M> * <mz>$ } in the local triad directions, where  $<M>$  is the value specified in the *Mass* field.

## D.8 External loads

Time- or state-dependent external loads can be updated based on the previous time instead of the current time, by specifying the description field command

```
#PrevStep
```

External loads in triads DOFs are applied in the defined system directions of the Triad, i.e., the same axis system boundary conditions are applied in, see [Section 5.5, "Triad properties"](#). This is normally equivalent to the global axis system. It is, however, also possible to let the loads act in the local with-rotated axis system independent of the chosen System directions, by specifying the command

```
#LocalAxes
```

in the description field of the load object. This is equivalent to the default behavior in Fedem R7.3.0 and earlier releases.



**TIP:** You can access the description field of a Triad DOF load by double-clicking the desired load entry in the Topology view of the actual triad.

## D.9 Sensors

Sensors on triads can measure rotations in terms of Rodriguez rotations by entering the command

```
#Rodrig
```

in the description field for the selected Sensor. The angular quantities measured by this sensor are then the component of rotation vector, as

defined in the Fedem 7.6 Theory Guide, Section 2.3.3 “Rodriguez parameterization”. The default is to measure Euler-Z-Y-X angles.

General functions using the default function argument, *Time*, can instead use the number of nonlinear iterations spent on the previous time step as argument, by specifying the following description field command:

```
#NumIt
```

## D.10 Wave functions

### D.10.1 Regular wave functions

Higher-order regular wave functions are defined from a regular Sine wave function through the following description field commands:

```
#Stokes5 - 5th order Stokes wave function
```

```
#Stream - Nonlinear streamline wave function
```

### D.10.2 Irregular wave functions

The frequency range of wave functions of type JONSWAP sea wave spectrum is calculated automatically or through user-defined period cut-off values specified in the property panel of the wave function. However, to specify the frequency interval explicitly, the following description field command can be used:

```
#OmegaRange <w0> <w1>
```

where  $<w0>$  and  $<w1>$  are, respectively, the lowest and the highest frequency in the wave spectrum.

### D.10.3 Embedded streamline functions

It is possible to blend in one (or several) nonlinear streamline wave function(s) into a irregular wave function defined from a JONSWAP spectrum. This is done through the following description field command:

```
#EmbeddedStream <r1> <r2> <t0> <Tp> <H>
```

- $r_1, r_2$  : Blending parameters (see equation below),  $0 < r_1 < r_2 < 1$
- $t_0$  : The time at which the embedded streamline wave should occur

- $T_p$  : Period of the embedded streamline wave function
- $H$  : Wave height of the embedded streamline wave function

The resulting wave elevation function is then established as follows:

$$\eta(x, t) = \frac{1}{2}(\eta_{JS}(x, t)(1 - w(t)) + \eta_{SL}(x, t)(1 + w(t)))$$

where  $\eta_{JS}(x, t)$  and  $\eta_{SL}(x, t)$  are the wave elevations of the background JONSWAP irregular wave and the streamline wave, respectively, whereas  $w(t)$  is a weighting (blending) function defined as:

$$w(t) = \cos\left(\left(\pi \frac{|t - t_0| - r_1 T_p}{(r_2 - r_1) T_p}\right), |t - t_0| \in [r_1 T_p, r_2 T_p]\right)$$

whereas  $w(t) = 1, |t - t_0| < r_1 T_p$  and  $w(t) = 1, |t - t_0| > r_2 T_p$ .

## D.11 Result output control

Simulating large models over long time series will generate a lot of results data. Users may then be interested in the results for only a few objects of a certain type, in order to reduce the size of the result database files. To facilitate such output selection, the options

`-allPrimaryVars-` and `-allSecondaryVars-` may be specified as additional solver options for the Dynamics Solver [Section 8.2.3](#), [\*"Additional solver options"\*](#) to switch off all output of primary and secondary solution variables, respectively. The following two description field commands are then provided to enable output for specific objects:

```
#savePos
```

This command enables saving of the position matrix for Triads, Parts and Beam objects in the model when `-allPrimaryVars-` is used. It has no effect for other object types.

```
#savevar <f1> <f2> ... <fn>
```

where the `<fi>` flags have the value 0 or 1 indicating whether a certain variable is to be saved for that object or not. This command is effective when the option `-allSecondaryVars-` is used. The number of flags in the `#saveVar` command varies depending on the type of object it is used for. The flags have the interpretation as shown in the following for the different object types.

### D.11.1 Triads

```
#savevar <f1> <f2> <f3> <f4> <f5> <f6> <f7>
```

- f1: Global velocity
- f2: Global acceleration
- f3: Global forces
- f4: Local velocity
- f5: Local acceleration
- f6: Local forces
- f7: Global deformations

### D.11.2 Parts

```
#savevar <f1> <f2> <f3>
```

- f1: Center of gravity
- f2: Generalized DOF components (displacement, velocity, acceleration)
- f3: Energies

### D.11.3 Beams

```
#savevar <f1> <f2> <f3>
```

- f1: Center of gravity
- f2: End sectional forces
- f3: Energies

### D.11.4 Joints

For joints, the `#savevar` flags are specified in groups of five, where each group is associated with the corresponding joint DOF.

```
#savevar <f1>1 <f2>1 <f3>1 <f4>1 <f5>1 <f1>2 <f2>2 <f3>2  

<f4>2 <f5>2 ... <f1>n <f2>n <f3>n <f4>n <f5>n
```

- $f_{1,i}$ : Deflection in joint DOF  $i$
- $f_{2,i}$ : Velocity in joint DOF  $i$
- $f_{3,i}$ : Acceleration in joint DOF  $i$
- $f_{4,i}$ : Friction/reaction force in joint DOF  $i$
- $f_{5,i}$ : Friction energy in joint DOF  $i$

The  $\langle f4 \rangle_i$  flag toggles the output of friction force if the joint DOF  $i$  is either Free or Spring-Damper constrained, and the reaction force if the joint DOF is either Fixed or Prescribed. To toggle output of the spring and/or damper forces in a spring-damper constrained joint DOF, you have to specify appropriate #savevar flags on the joint spring and damper objects as given in [Section D.11.5, "Springs"](#) and [Section D.11.6, "Dampers"](#), respectively. To access the description fields of these objects, right-click the desired joint spring/damper object in the *Topology* view and choose **Select**, see [Section 2.4.4, "ID and Topology panel"](#).

## D.11.5 Springs

```
#savevar <f1> <f2> <f3> <f4> <f5>
```

- f1: Spring stiffness
- f2: Spring length
- f3: Spring deflection
- f4: Spring force
- f5: Energies

## D.11.6 Dampers

```
#savevar <f1> <f2> <f3> <f4> <f5>
```

- f1: Damping coefficient
- f2: Damper length
- f3: Damper velocity
- f4: Damping force
- f5: Energies

## D.11.7 Loads (Force and Torque)

```
#savevar <f1> <f2> <f3>
```

- f1: Global force/torque vector
- f2: Signed force/torque amplitude
- f3: Energies

### D.11.8 Mechanism

```
#savevar <f1> <f2> <f3>
```

- f1: Center of gravity
- f2: Energies
- f3: Algorithm parameters



**NOTE:** The #savevar command of the Mechanism object is entered in the Model description field of the Model Preferences dialog box, see [Section 4.9, "Model preferences"](#).

## D.12 Undocumented features

In addition to the description field commands discussed in the above sections, there also exists another set of commands without any further documentation. They are used for features that are still under testing, or as a “back-door” for some special-purpose features. For the sake of completeness, these commands are listed in the table below.



**WARNING!** Do not use any of these commands unless you know what you are doing. If you encounter a Fedem model file where one or more of them are being used, please consult the Fedem support team before using that file.

Command	Object Type	Comment
#Displace	FE Part	Support for sub-model analysis
#LocalDofs	Free Joint	
#RefTriads	FE Part	
#TetGen	FE Part	
#BALL_FRICTION #BALL_FRICTION2	Rotational Friction	
#EndTime	Mechanism	Applied in Model Preferences
#ModalDamping	Mechanism	Applied in Model Preferences
#waveTheory	Mechanism	Applied in Model Preferences
#Drag	Generic Part	
#Slam	Generic Part	
#Bodygroup	Generic Part	
#DragTX #DragTY #DragTZ #DragRX #DragRY #DragRZ	Generic Part	

Command	Object Type	Comment
#PrintSupelDef	FE Part	
#Fixed	Part	
#Projection	FE Part	
#Version	Joint (any type)	
#BallFriction	Ball Joint	
#PipeRadius #OuterPipeRadius	Friction	DrillSim support
#HydroFric #SkinFric #RadFric	Friction	DrillSim support
#ShowDir	Triad	
#DragTX #DragTY #DragTZ	Triad	
#FlipWCaxis	Tire	
#Old	Wave function	User-defined wave spectrum
#ramp	General function	Only for function type <i>Constant</i>
#Params	User-def. element	
#Property	User-def. element	
#Engine	User-def. element	
#Morison	User-def. element	
#Wear	Graph	
#skipBlade	Animation	
#Model	Graph	
#PVX	Curve	
#noClip	Curve	

# Index

## A

absolute integration tolerances, 8-24  
active view, 2-24  
    enlarge, 2-25  
    reduce, 2-25  
    scaling, 2-25  
active window  
    controlling, 2-7  
adder block, 7-4  
additional boundary conditions for triads, 8-27  
additional solver options, 8-8, C-1  
Align CS, 4-10  
Align rotations, 4-10  
Altair® HyperMesh®, 1-4  
amplifier block, 7-4  
amplifiers, 7-4  
analysis options  
    management of, 8-7  
animation  
    available results, 9-35  
    control, 9-38  
    loading performance, 9-37  
    properties, 9-29  
Animation Control, 9-41  
Animation Control panel, 9-40  
animation frames, 9-31, 9-35  
    load fringes and deformations, 9-31  
    stress recovery options, 9-31  
animation speed, 9-39  
animations, 9-2  
    averaging options, 9-32  
    close, 9-39  
    cycling, 9-39  
    deformation results, 9-29  
    eigenmode, 9-33  
    eigenmodes tab, 9-29  
    fringes tab, 9-29

loading, 9-29  
Modeler window, 9-27  
Play Panel, 9-27  
playing repeatedly, 9-39  
showing continuous motion, 9-31  
specifying averaging behavior, 9-33  
speed, 9-39  
time tab, 9-29  
anti-aliasing, 2-29  
ASCII format, 2-38, B-2  
attached elements  
    color, 4-21  
    *See also* general appearance  
autoscale, 9-7  
averaging options, 9-32  
axial dampers, 5-56  
axis rotation, 2-22

## B

Back view, 2-23  
Backward Euler, 8-2  
ball joint, 5-41  
ball movement, 4-11  
batch-execution of solvers, C-1  
.bdf, 2-3  
.bdf format, 1-4, A-1  
binary format, B-2  
binary input blocks  
    adder, 7-4  
    comparator, 7-4  
    multiplier, 7-4  
binary-input control elements, 7-4  
.bmp format, 2-38  
Bottom view, 2-23  
Bulk Data File format, 1-4  
    exporting to, 1-4  
    *See also* .bdf

## C

CAE systems, A-1  
cam joint, 5-46  
    creating, 5-47  
    follower, 5-46  
cam surface, 5-46  
Cam thickness, 5-49  
cam triads, 5-46  
Centre of Gravity, 5-3  
centripetal moment correction, 8-25  
co-located items, 2-20  
Color mapping, 9-40  
colors for attached/unattached elements, 4-21  
command sensitivity, 2-7  
commands  
    accessing, 2-8  
    delete all stickers, 4-7  
    detach, 4-21  
    general appearance, 2-26  
    item appearance, 2-29  
    modeling, 2-8  
    multiple selection, 2-19  
    observing with Output List, 2-18  
pan (F1), 2-22  
    performing, 2-18  
rotate (F3), 2-22  
    select dynamic center (F4), 2-22  
3D viewing, 2-24  
    zoom (F2), 2-22  
comparator block, 7-4  
complex conjugate pole block, 7-7  
Component Mode Synthesis, 8-2, 8-14  
component modes, 8-14  
    calculating, 8-14  
connections  
    stiff translating, 5-44  
    triad, 4-2, 5-26  
contact, 5-49 — 5-50  
control blocks (control elements), 7-4  
    about, 7-4  
    controlling output, 7-6  
    creating and manipulating, 2-16

Control Creation toolbar, 2-8, 7-2  
Control Editor, 1-3, 2-16, 7-2  
    deleting blocks or connections, 7-10  
    opening, 2-16, 7-2  
Control menu, 7-2  
control module  
    defining, 7-7  
control modules  
    building, 7-2  
control system  
    about, 7-1  
    amplifiers supported, 7-4  
    connection with mechanism, 7-3  
    creating, 7-2  
    diagram of, 2-16  
    modeling, 1-3, 7-1  
control system blocks  
    adding lines, 7-9  
    defining the module, 7-9  
    editing properties, 7-8  
    inserting, 7-8  
    moving, 7-8  
    removing breaking points, 7-9  
    rotating, 7-10  
control toolbars, 7-2  
control tools, 2-8  
Control Tools toolbar, 7-3, 7-8  
control/servo systems, 1-3  
coupling effects, 1-3  
Ctrl key  
    using, 4-21  
curves  
    exporting, 2-38 — 2-39  
    modifying appearance, 9-16  
cut-back, 8-23  
cylindric joint, 5-45  
cylindrical motion, 4-11

## D

.dac format, 2-38  
dampers  
    properties, 5-55

---

dead-zone block, [7-6](#)  
lower and upper limits, [7-6](#)  
output, [7-6](#)

degrees of freedom (DOFs), [5-64](#)

delay block, [7-5](#)

delete  
    all stickers, [4-7](#)  
    selecting multiple items, [2-19](#)

deleting  
    stickers and recreating, [4-7](#)

derivator block, [7-5](#)

description field commands, [D-1](#)

deselecting items in the Model Manager, [2-9](#)

detach, [4-21](#)

DOFs, [5-31](#), [5-64](#)  
    *See also* degrees of freedom

dynamic  
    functions, [2-21](#)  
    pan (F1) command, [2-22](#)  
    rotate (F3) command, [2-22](#)  
    select dynamic center (F4)  
        command, [2-22](#)  
    viewing, [2-21](#)  
    zoom (F2) command, [2-22](#)

dynamic zooming and rotation, [2-22](#)

dynamics analysis  
    about, [8-2](#)  
    controlling parameters, [8-21](#)  
    performing, [8-2](#)  
    unbalanced forces, [8-3](#)

dynamics simulation, [10-9](#)  
    before beginning, [8-3](#)  
    deleting result files, [8-52](#)  
    result files, [10-9](#)  
    starting, [8-38](#)

Dynamics Solver  
    option files, [10-9](#)  
    setup, [8-21](#), [8-38](#)

**E**

eigenmode  
    specify shift factor, [8-27](#)

Eigenmode animations, [9-30](#)

eigenmode solutions, [1-2](#)  
    calculating, [8-27](#)  
    computed by Fedem, [8-4](#)

eigenmodes  
    calculating, [8-4](#)  
    damped, [8-27](#)  
    options, [9-34](#)  
    specify number to be computed, [8-27](#)

element  
    expressions syntax, [A-5](#)  
    properties, [1-4](#)  
    topologies, [1-4](#)

Element group properties, [5-4](#), [5-19](#), [5-22](#)

element groups  
    creation, [5-6](#), [5-18](#)  
    other identifiers, [A-12](#)

element statements  
    parameters, [A-4](#)

equilibrium analysis  
    conditions, [8-29](#)

error messages  
    intepretation, [8-54](#)

example control system, [7-2](#)

exporting  
    3D modeler view, [2-40](#)  
    animations, [2-38](#), [2-41](#)  
    curves, [2-38](#)  
    graphs, [2-38](#), [2-40](#), [9-22](#)  
    links, [2-38](#)  
    objects, [2-38 — 2-39](#)

external nodes, [1-4](#)

**F**

F1 (dynamic pan), [2-22](#)  
F2 (dynamic zoom), [2-22](#)  
F3 (dynamic rotate), [2-22](#)  
F4 (select dynamic center), [2-22](#)

Fedem, [2-7](#)  
    about, [1-2](#), [2-1](#)

---

binary input blocks, [7-4](#)  
building techniques, [4-2](#)  
commands, [2-1](#)  
control systems, [7-2](#)  
    available control elements, [7-4](#)  
controllers, [7-6](#)  
definition of, [1-2](#)  
directory structures, [2-2, 10-8](#)  
editing environment, [7-2](#)  
file formats, [B-2](#)  
file types, [2-2](#)  
    input, [B-2](#)  
    intermediate, [B-2](#)  
    results, [B-2](#)  
    secondary, [B-2](#)  
file use overview, [B-4](#)  
files and directory structures, [B-1](#)  
graphical representation, [7-2](#)  
main window, [2-6](#)  
major user tasks, [4-3](#)  
mechanism, [1-4](#)  
    analysis, [8-1](#)  
    elements, [3-1, 4-1](#)  
model  
    definition of, [1-4](#)  
modeling environment, [4-2](#)  
module execution, [1-5](#)  
reducer  
    options, [C-2](#)  
results database  
    removing unwanted files, [10-9](#)  
starting, [2-3](#)  
strain rosette analysis, [1-6](#)  
technical support, [2-5](#)  
user interface, [2-1, 2-6](#)  
Fedem Link Model ([.f1m](#)) format, [A-1](#)  
Fedem Mechanism Model ([.fmm](#))  
format, [2-2](#)  
Fedem Technology Link ([.ft1](#))  
format, [A-1 — A-2](#)  
    nodes, [A-3](#)  
file types  
    input files, [B-2](#)  
intermediate files, [B-3](#)  
other files, [B-3](#)  
results files, [B-3](#)  
Finite Element (FE)  
    dynamics in elastic mechanisms, [1-2](#)  
    material data, [1-4](#)  
    method, [1-5](#)  
    models, [1-4](#)  
        creating, [1-4](#)  
        generation of, [A-1](#)  
        importing, [1-4](#)  
        nonlinear, [1-4](#)  
        storage, [2-3, A-1](#)  
        *See also* links  
    nodes, [1-4](#)  
first-order transfer function block, [7-7](#)  
five-DOF joint, [5-51](#)  
flip element direction, [7-10](#)  
.f1m format, [2-3, A-1](#)  
.fmm format, [2-2](#)  
follower triads  
    constraining, [5-47](#)  
force vector, [5-61](#)  
force vector orientation, [5-61](#)  
forces, [5-60](#)  
format  
    .f1m, [2-3](#)  
    .fmm, [2-2](#)  
    .ft1, [2-3](#)  
free joint, [5-42](#)  
free movement, [4-11](#)  
friction, [5-34, 5-41 — 5-42, 5-45, 5-49](#)  
    behavior, [5-53](#)  
    computing force, [5-53](#)  
    creating, [5-53](#)  
    editing, [5-53](#)  
    parameters, [5-53](#)  
    properties, [5-53](#)  
    with appropriate joint, [5-53](#)  
fringe legend, [9-40](#)  
    customizing, [9-40](#)  
fringe value  
    selecting results, [9-32](#)

From and To options, 5-61  
front view, 2-23  
.f1 files, 5-28  
examples, A-2  
identifiers, A-2  
.f1 format, 2-3, 2-38, A-2  
Full Color, 9-41  
Full Color B/W Limits, 9-42  
Full Color Clipped Limits, 9-42  
function  
    creating, 5-63  
function keys, 2-21  
functions  
    properties, 5-63

## G

general appearance, 2-26  
general transfer functions, 7-7  
geometric stiffness contribution (link stiffening), 8-25, 8-27 — 8-28  
geometrically nonlinear FE model  
    treatment of, 1-4  
graphic performance  
    increasing speed, 2-24, 2-28  
    modeling tools, 1-3  
graphics card settings, 2-29  
graphs  
    about, 9-3  
    abscissa label, 9-4  
    adding legend, 9-4  
    controlling x- and y-axis range, 9-7  
    creating, 9-4  
    creating curves, 9-4  
    displaying, 9-4  
    displaying curve properties, 9-7, 9-23  
    dynamic updating, 8-51  
    dynamic viewing, 9-19  
    enabling Autoscale option, 9-4  
    exporting, 2-38  
    naming, 9-4  
    naming curves, 9-8  
    opening window, 2-17

ordinate label, 9-4  
Possible Results list, 9-10, 9-12, 9-15  
printing, 9-22  
result operation, 9-9  
selecting results, 9-10, 9-24  
specifying properties, 9-4  
variables, 9-10  
viewing tips, 9-19  
viewing values, 9-3  
views, 2-7, 2-17, 8-51, 9-2  
    displaying legend, 9-8  
    manipulating, 9-19  
    updating curves, 8-51  
X-Axis/Y-Axis, 9-8  
Gravitation, 4-24  
grid and snap, 2-16  
    manipulating, 7-8  
ground  
    selecting, 2-20  
group-wise solving, 8-12  
Guide bar, 2-19

## H

hardware requirements, 2-2  
HyperMesh®, 1-4  
hysteresis (backlash) block, 7-6  
    deadband, 7-6

## I

ID and Topology panel, 2-7, 2-10  
    sensors, 5-77  
ID Number, 2-10  
I-DEAS®, 1-4  
identification numbers, 2-9  
importing links, 2-8  
initial equilibrium  
    analysis, 5-27, 8-28  
Initial translational velocity, 4-24  
input and output blocks, 7-3  
input file types, B-2  
input force orientation, 5-62  
input signal

- amplifying, 7-4  
calculating and outputting power, 7-4  
integrator and limited derivator  
blocks, 7-5  
integrator block, 7-5  
Interactive Odometer, 4-5, 4-8  
about, 4-5  
specifying a discrete point, 4-12  
using with Smart Move command, 4-5  
intermediate file types, B-3  
isometric button, 2-23  
item type, 2-10  
iteration step size, 8-29  
Iteration step size limit, 8-29
- J**
- joint  
variables, 5-33  
joint pairs, 5-51  
Joint Variables, 5-31  
joint variables, 5-33  
joints  
about, 5-31  
adding friction, 5-34  
adding motion constraints, 5-36  
attaching, 4-17  
attaching multiple, 4-17  
attaching to links, 5-31  
attachment restrictions, 4-17  
ball, 5-40  
cam, 5-46  
cylindric, 5-45  
detaching, 4-21  
free, 5-40  
friction, 5-53  
master and slave, 4-17  
pair constraints  
rack and pinion, 5-51  
point-to-path, 5-44  
point-to-point, 5-40  
prismatic, 5-44  
revolute, 5-40
- rigid, 5-40  
summary table, 5-37  
triad connections, 5-31
- L**
- left view, 2-23  
License, 2-42  
limiter block, 7-6  
line view, 2-24  
lines  
definition, 7-10  
lines (item appearance), 2-30  
Link coordinate system, 5-3  
Link Database, 5-17  
link files  
directory, 10-1  
Link\_DB directory, 2-2  
subdirectories, 10-9  
link\_DB directory, 10-9  
links, 4-8  
about, 5-2  
applied forces, 5-60  
as masters and slaves, 5-31  
changing appearance, 2-29  
duplicating, 5-7  
exporting, 2-38 — 2-39  
importing, 2-8, 5-5, 6-4, 6-11  
importing several at once, 5-5  
proportional damping, 5-9  
link-wise solving, 8-11  
load fringes, 9-29  
loads, 5-60  
load target point, 5-61  
loads and torques  
control of magnitude, 5-60  
logical-switch block, 7-5  
lumped mass matrix, 8-14
- M**
- mass and stiffness matrices, 8-2  
master/slave triad, 5-32  
attaching manually, 5-32

---

MATLAB®/Simulink®, 1-3  
MATRIXx®, 1-3  
mechanism, 1-4  
    3D view of, 2-15  
    accessing a response variable, 7-3  
    building, 4-4  
    components  
        flexibility of, 1-3  
        element connections, 1-4  
    introducing motion, 5-60  
    modeling process, 4-1  
mechanism analyses, 2-7  
    speeding up, 8-2  
Mechanism Creation toolbar, 4-4  
mechanism elements, 3-1, 4-1  
    attaching to a link, 4-16  
    constrained by cylindrical joint, 4-11  
    constrained by prismatic joints, 4-11  
    creating, 2-8, 4-8  
    customizing, 5-1  
    defining properties, 5-1  
    deleting from model, 4-22  
    detaching, 4-21  
    fully constrained, 4-11  
    limiting display of, 2-20  
    links, 4-8  
    measuring movement and  
        variables, 5-76  
    moving, 3-36, 4-10  
    viewing and editing properties, 2-11  
mechanism entities  
    display manipulation, 2-28  
mechanism mode shapes  
    animating and displaying, 8-40  
mechanism model  
    building, 4-1, 4-3  
mechanism symbols, 2-27  
    manipulating appearance, 2-26  
Mechanism Tools toolbar, 4-4  
menus and toolbars, 2-6  
modal analysis, 8-4  
mode and time lists, 8-41  
mode shape analysis, 8-4 — 8-5  
    setting up, 8-40  
    specifying parameters, 8-4 — 8-5  
mode shape recovery, 8-41  
eigenmodes animations, 9-33  
options, 8-41  
mode shapes  
    animating, 8-40 — 8-41  
    expanding, 8-4  
model  
    applying motion constraints, 4-1  
    reduction process, 1-4  
    storing contents, 2-2  
    visualizing, 2-21  
model display  
    color, 2-29  
    complexity level, 2-29  
    fog, 2-28  
    transparency, 2-29  
model file  
    creating new, 2-4  
    directory, 2-2  
Model Manager  
    selecting items, 2-9  
Model Manager Objects list, 4-7  
Model Manager panel, 2-6, 2-9  
Model Manager Results list, 9-2  
Model Manager tabs, 4-2  
model objects  
    editing properties, 2-7  
    managing, 2-6, 2-9  
model reduction, 8-2, 8-13  
    initiating manually, 8-13  
Modeler window, 2-15, 4-3  
    Global Directions, 2-16  
    Interactive Odometer, 4-8  
    locating specific points, 4-5  
    manipulating the view, 2-7  
    opening, 4-3  
    Reference Plane, 2-16  
modeling  
    aids  
        stickers, 4-6  
    commands, 2-8

---

control systems, 1-3  
objects  
    triads, 5-26  
programs, 1-4  
tools, 1-3, 4-4  
Modeling tolerance, 4-24  
modeling tools, 4-3  
models, 2-28  
models and results  
    constructing and viewing, 2-6  
modes  
    postprocessing, 8-41  
modified Newton-Raphson  
iteration, 8-24  
motion  
    constraints, 2-8, 4-10  
mouse  
    using, 2-19  
Mouse button, 5-61  
movability, 4-4, 4-6  
movability, 4-2  
move curves, 9-6  
MSC/Patran®, 1-4  
MSC.Nastran®  
    Bulk Data File (.bdf) format, 1-4  
multibody systems  
    applications, 1-3  
multiple face results, 9-33  
multiplier block, 7-4

attributes, 1-4  
coordinates, 1-4  
nodes  
    syntax for, A-3  
nonlinear equations  
    solution of, 8-24

## O

object movability  
    determining, 4-10  
object selection, 2-18  
    confirming, 2-20  
    ground, 2-20  
    restrictions, 2-20  
objects  
    related, 2-11  
    selection history, 2-19  
Objects list  
    functions, 5-63  
Objects tab, 2-10  
option files  
    storage, 10-9  
Origin property, 4-10, 4-14  
output block  
    definition, 7-3  
Output List  
    attachment process, 4-16  
    window, 2-18  
        opening, 2-18

## N

.nas format, 2-3, A-1  
Nastran Bulk Data (.nas or .bdf)  
format, A-1  
navigating, 2-13  
nCode DAC (.dac) format, 2-38  
Negative pivots, 8-16  
Newmark time-integration, 8-2  
Newton integration algorithm, 8-21  
Newton-Raphson equilibrium  
iterations, 8-2  
nodal

pan (F1), 2-22  
pan down, 2-26  
pan left, 2-25  
pan right, 2-26  
pan up, 2-26  
panels  
    hiding, 2-16  
parallel projection, 2-24  
perform, 1-5  
perspective view, 2-24  
PI, PD, and PID controllers, 7-6

PID control block, 7-9  
pin joints, 5-39  
Play Panel, 9-39  
point rotation, 2-22  
point-force vectors, 5-60  
point-to-path joints, 5-44 — 5-45  
point-to-point joints, 5-40  
polygons (item appearance), 2-29  
postprocessing  
    animating options, 9-2  
    capabilities, 1-2  
    definition, 9-2  
    graphing options, 9-2  
    modes, 8-41  
power block, 7-4  
prismatic and cylindric joints, 5-45  
    improving load distribution, 5-45  
prismatic joints, 5-44  
    adding friction, 5-45  
Pro/ENGINEER®, 1-4  
processing results  
    deleting, 8-51  
    storing, 8-51  
program modules, 1-5  
    strain rosette analysis, 1-6  
properties, 2-12  
Property Editor panel, 2-7, 2-11  
Property menues, 2-12

## R

rack-and-pinion, 5-51 — 5-52  
    transmission ratio, 5-52  
Radial contact, 5-50  
rainflow analysis, 8-49  
real-pole block, 7-7  
recovery operations  
    deleting result files, 8-52  
Red Blue, 9-42  
reducer  
    options, C-2  
Reference, 2-29  
Reference Plane, 4-4

changing appearance, 2-29, 4-4  
relations, 2-11  
relative sensor, 5-77  
remove breakpoint, 7-10  
resolution settings, 2-2, 2-37, 2-45  
result directories, 10-10  
result set  
    by name, 9-32  
    by operation, 9-32  
results file types, B-3  
results files, 10-9  
Results list  
    managing, 9-2  
    shortcut menus, 9-2  
Results tab, 2-10  
revolute joint, 5-40  
.rgb format, 2-38  
Right view, 2-23  
rigid body animation, 8-40 — 8-41, 8-51  
rigid joint, 5-41  
rigidity symbol, 4-11  
Road, 5-65  
rotate (F3), 2-22  
rotating  
    about a point, 2-22  
    about an axis, 2-22  
    selecting new point, 2-23  
rubber bushings, 5-39  
Run... option, C-1  
Runge-Kutta method (Lobatto IIIC), 8-2

## S

sample-and-hold block, 7-5  
Save As... command, 8-52  
screw ratio, 5-46  
second-order accuracy  
    achieving, 8-2  
second-order transfer function  
block, 7-7  
Select, 2-22  
select, 2-13  
select dynamic center (F4), 2-22

---

selecting items in the Model Manager, 2-9  
selection, 2-19  
selection filter, 2-20  
selection history, 2-19  
sensors, 5-76  
    managing, 5-77  
    processing data, 5-77  
    simple, 5-77  
Show All Frames forcing option, 9-39  
Simplified visualization, 5-3  
simulating nonlinear behavior, 7-5  
simulation  
    graphed results, 9-3  
    managing results, 9-2  
    progress indication, 8-51  
simulation results  
    dependent on, 1-5  
singularities, 8-15  
Singularity criterion, 8-16  
slave triad (follower), 5-46  
Smart Move, 4-5, 4-8, 4-10  
    from-point and to-point, 4-11  
    motion constraints, 4-6  
    performing, 4-11, 4-13  
    selecting multiple items, 2-19  
    stickers, 4-6  
    types of motion allowed, 4-10  
    using, 4-6  
S-N curve, 5-20  
snap, 2-19  
solid view, 2-24  
solution processes  
    stopping, 8-51  
solvers  
    command-line options, C-1  
    running in batch mode, 10-9  
    setting up parameters, 8-8  
Solvers toolbar  
    about, 8-7  
    accessing commands, 8-8  
speed slider, 9-39  
spring inter-connectivity, 5-38  
springs  
    assigning values, 5-32  
    changing stress-free length during simulation, 5-55  
    properties, 5-54  
static equilibrium analysis  
    about, 8-3  
    setting up, 8-3  
Status bar, 2-7  
stickers, 4-6, 4-8  
    applying manually, 4-6  
    creating manually, 4-7  
    deleting, 4-7  
storage, 2-2  
    eigenvalues from dynamics simulation, 10-10  
Fedem Link Model (.f1m) format, 2-2  
Fedem Mechanism Model (.fmm) format, 2-2  
Fedem Technology Link (.ft1) format, 2-2  
link files, 2-2  
MSC.Nastran Bulk Data File (.bdf or .nas) format, 2-2  
primary time history result files, 10-10  
results from eigenvalue recovery, 10-10  
results from stress recovery, 10-10  
simulation results, 2-2  
strain coat analysis, 8-47  
    options, 8-48  
    recovery, 8-49  
strain coat elements, 8-47  
strain rosette analysis, 1-6, 8-42  
    options, 8-42  
    result files, 8-43  
strain rosette definition, 8-45  
stress analysis, 1-2  
    improving performance, 8-38  
    specifying parameters, 8-4 — 8-5, 8-38  
Stress concentration factor, 5-20  
stress recovery

analysis, 8-4  
starting, 8-40  
superelements, 1-4, 8-2  
symbols  
line-smoothing, 2-29  
system requirements, 2-1 — 2-2  
*See also* hardware requirements  
system resources, 8-51

## T

target point, 5-61  
3D  
animation, 8-51  
modeling  
stickers, 4-10  
point marker, 4-5  
viewing, 2-15  
dynamic updating, 8-51  
3D View Control toolbar (View menu), 2-24  
3D viewing, 2-27  
commands, 2-24  
time history  
changing interval, 9-30  
Time History Animations, 9-30  
time integration  
optimizing numerical performance, 8-24  
time step iterations, 8-51  
Time summary animations, 9-30  
time window, 9-31  
time-dependent control blocks  
delay, 7-5  
sample-and hold, 7-5  
Tire model, 5-65  
tool, 2-8  
toolbars  
3D View Control, 2-7  
Control Creation, 2-8, 7-2  
Control Tools, 2-8, 7-3  
managing, 2-8  
mechanism tools, 2-7

Solvers, 2-7  
standard, 2-7  
Windows, 2-7  
Zoom and Pan, 2-7  
top view, 2-23  
Topology List, 2-10  
torques, 5-60  
translational motion  
restraining, 4-6  
triads, 4-2, 5-2, 5-26  
about, 5-26  
adding mass and mass inertias, 5-28  
additional boundary conditions, 5-28  
attachment restrictions, 4-17  
color representations, 5-27  
connections, 4-2, 5-26  
constrained DOFs, 5-31  
coordinate system, 5-27  
editing properties, 5-28  
FE node, 5-28  
follower, 5-46  
in joints, 4-17, 5-26  
master and slave, 5-31  
purposes, 5-27  
restraining movement, 5-28  
symbols, 5-27  
triangularization, 8-15

## U

unattached elements  
color, 4-21  
*See also* general appearance  
undo option, 4-7  
Units, 4-24  
user interface, 2-1

## V

variables  
functions of, 7-3  
view  
achieving maximum zoom, 2-22  
axis rotation, 2-22

---

bottom, 2-23  
flat colors, 2-24  
front, 2-23  
isometric, 2-23  
magnifying, 2-23  
manipulating, 2-21  
parallel, 2-24  
perspective, 2-24  
point rotation, 2-22  
rotate, 2-22  
rotating, 2-7  
selecting dynamic center, 2-22  
show top faces, 2-24  
side, 2-23  
solid/shaded objects, 2-24  
top, 2-23  
View button, 5-61  
viewer options, 2-28  
viewing capabilities, 2-15  
views

animations, 9-2  
graphs, 8-51, 9-2

## W

Workspace, 2-6, 2-15  
displaying results, 9-2  
graph views, 2-17  
managing windows in, 2-15

## Z

zoom (F2), 2-22  
zoom all, 2-25  
Zoom and Pan toolbar (View menu), 2-24, 9-19  
zoom in, 2-25  
zoom out, 2-25  
zoom window, 2-25

[REDACTED]

---

---

[www.sap.com/contactsap](http://www.sap.com/contactsap)

©2020-2022 SAP SE or an SAP affiliate company. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company. The information contained herein may be changed without prior notice.

Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies.

Please see <https://www.sap.com/about/legal/trademark.html> for additional trademark information and notices.

**THE BEST RUN** 