

# Vehicle Modeling and Simulation using Adams/Car

ADM740 Course Notes

*September 2017*

## Legal Information

MSC Software Corporation reserves the right to make changes in specifications and other information contained in this document without prior notice. The concepts, methods, and examples presented in this text are for illustrative and educational purposes only and are not intended to be exhaustive or to apply to any particular engineering problem or design. MSC Software Corporation assumes no liability or responsibility to any person or company for direct or indirect damages resulting from the use of any information contained herein.

Copyright © 2017 MSC Software Corporation. All Rights Reserved. This notice shall be marked on any reproduction of this documentation, in whole or in part. Any reproduction or distribution of this document, in whole or in part, without the prior written consent of MSC Software Corporation is prohibited.

The MSC Software corporate logo, Adams, Dytran, Easy5, Fatigue, Laminate Modeler, Marc, Mentat, Patran, MSC, MSC Nastran, Mvision, Patran, SimDesigner, SimEnterprise, SimManager, SimXpert and Sofy are trademarks or registered trademarks of the MSC Software Corporation in the United States and/or other countries. NASTRAN is a registered trademark of NASA. All other trademarks belong to their respective owners.

# CONTENTS

Section	Page
<b>0 Introduction</b>	
Welcome to Adams/car training	0-2
What is Adams?	0-4
What is Motion Simulation?	0-5
SimCompanion	0-8
Getting Help	0-14
<b>1 Introducing Adams/Car</b>	
Motivation for Using Adams/Car	1-4
User Modes	1-8
Database Structure – A Directory Hierarchy	1-9
Saving Files: Working Directory Versus Database	1-11
Configuration Files	1-12
Workshop 1: “Open and Run an Assembly”	
<b>2 Basic concepts</b>	
Data Hierarchy	2-6
Test Rig	2-8
Major and Minor Roles	2-11
Naming Convention	2-13
Workshop 2: “Template Versus Subsystems”	
<b>3 Creating and Adjusting Subsystems</b>	
Creating Subsystems	3-4
Modifying Subsystems	3-5
Adjusting Hardpoints	3-6

# CONTENTS

Section	Page
<b>3 Creating and Adjusting Subsystems (Cont.)</b>	
Adjusting Parameter Variables	3-8
Adjusting Mass Properties	3-10
Adjusting Springs and Dampers	3-11
Curve Manager and Property File Editor	3-15
Replacing Instance Definition	3-24
More Subsystem Adjustments	3-25
Workshop 3: "Creationg and Adjusting Subsystems"	
<b>4 Creating and Simulating Suspensions</b>	
Creating Suspension Assemblies	4-4
Half-Vehicle Analyses	4-6
Suspension Parameters	4-8
Creating Loadcases	4-10
Dynamic Analysis	4-11
Warning Messages	4-12
Files Produced by Analyses	4-14
Plot Configuration Files	4-15
Workshop 4: "Running Suspension Analysis"	
<b>5 Importong CAD Geometry</b>	
Importing CAD Geometry	5-2
Workshop 5: "Importing CAD Geometry"	



# CONTENTS

Section	Page
<b>6 Creating and Simulating Full Vehicles</b>	
Creating Full-Vehicle Assemblies	6-4
Shifting Subsystems	6-6
Updating Subsystems and Assemblies	6-7
Synchronizing Subsystems and Assemblies	6-8
Adding/Removing and Activating/Deactivating Subsystems	6-9
Adjusting Mass	6-12
Static Vehicle Set-up	6-14
Workshop 6: "Running Full-Vehicle Analysis and Adjusting Mass"	
<b>7 Driving Machine</b>	
Standard Driver Interface (SDI) and Driving Machine	7-4
Using the Driving Machine	7-5
Data Flow	7-7
Event Files	7-8
Example Event Files	7-9
Driver Control Data Files	7-10
Creating .DCD Files	7-12
Workshop 7: "Creating Event Files"	
<b>8 Tires</b>	
Tire Overview	8-4
Handling Tire Models	8-6
Road Types	8-7
Tire-Road Contact Methods	8-9

# CONTENTS

Section	Page
<b>8 Tire (Cont.)</b>	
Special Tire Models	8-12
Adams/Tire Tools	8-13
Which Tire Model Should You Use?	8-14
Overview of the Adams/Tire Model Features	8-17
How You Use Adams/Tire	8-18
Workshop 8: "Tire Testrig Tutorial"	
<b>9 Road Builder</b>	
Road Types	9-4
3D-Spline Road Property File Structure	9-9
Why Use Road Builder?	9-12
Using the Road Builder	9-13
Workshop 9: "Creating Road Property Files using Road Builder"	
<b>10 Using Flexible Bodies</b>	
Flexible Body Overview	10-4
Limitations of Flexible Bodies	10-6
Getting Flexible Bodies	10-7
Modal Superposition	10-8
Visualization Attributes	10-10
About Joints and Motions	10-15
Workshop 10: "Flex Tutorial"	
<b>11 General Actuation Analysis</b>	
GAA Model Details	11-5

# CONTENTS

Section	Page
<b>11 General Actuation Analysis(Cont.)</b>	
Accessing Actuator Parameters	11-6
Using Request Map Editor	11-8
Opening/Saving Actuation Input Map File	11-10
General Actuation Parameters	11-13
Workshop 11: "General Actuation Analysis"	
<b>12 Building Templates</b>	
Parameterization in Adams/Car	12-5
Creating Hardpoints	12-6
Creating Construction Frames	12-8
Location Parameterization	12-9
Orientation Parameterization	12-14
Template Overview	12-24
Template Topology	12-25
File Architecture	12-26
Template Components	12-28
Types of Parts	12-29
Geometry	12-31
Attachments (Joints and Bushings)	12-32
Springs	12-33
Dampers	12-35
Bumpstops and Reboundstops	12-36
Toe/Camber Angles and Suspension Parameter Array	12-38

# CONTENTS

Section	Page
<b>12 Building Templates (Cont.)</b>	
Adjustable Forces	12-40
Parameter Variables	12-41
General Advice	12-42
Workshop 12: "Template-Builder Tutorial"	
<b>13 Communicators</b>	
Types of Communicators	13-4
Classes of Communicators	13-6
Communicator Symmetry	13-9
Communicator Roles	13-10
Naming Communicators	13-11
Matching Communicators During Assembly	13-13
Matching Communicators With Test Rigs	13-18
Workshop 13: "Getting Information about Communicators"	
<b>14 Requests</b>	
Creating New Requests	14-4
Types of Requests	14-5
Toggle Request Activity	14-9
Workshop 14: "Requests"	
<b>15 Exploring Templates</b>	
Investigating Templates	15-4
About the Database Navigator	15-7
Workshop 15: "Exploring and Completing Templates"	

# CONTENTS

Section	Page
<b>16 Additional Applications</b>	
Adams/Car Ride	16-4
Adams/Vibration	16-8
Adams/Durability	16-11
Adams/Driveline	16-13
Adams/Linear	16-16
Adams/Controls	16-17
Adams/Insight	16-18
Workshop 16: "Linear Modes Analysis"	
<b>17 Tilt Table Analysis</b>	
Tilt Table Analysis	17-2
Adams/Car Truck Plugin	17-5
Setting up the Analysis	17-6
Inputs to the Analysis	17-7
<b>18 Using Adams/Insight with Adams/car</b>	
What is Experimental Design?	18-4
Response Surface Overview	18-5
Response Surface Equations	18-6
Factor (Parameter) Definition	18-7
Response (Objective) Definition	18-9
Typical Samples Required for a Cubic Fit	18-10
Response Surface Equation and Quality	18-12
Pareto Chart Output	18-13

# CONTENTS

Section	Page
<b>18 Using Adams/Insight with Adams/car (Cont.)</b>	
Starting Adams/Insight from Adams/Car	18-14
Fitting Results	18-15
Publishing Results	18-16
Workshop 18: "Using Adams Insight with Adams Car"	
<b>19 SVC and SPMM Events in Adams car</b>	
SVC Overview	19-4
Suspension – SVC	19-5
Suspension – SVC Output	19-6
Full Vehicle SVC	19-7
Sprung Mass Table	19-8
Full Vehicle SVC Output	19-9
SPMM	19-10
SPMM Sub-Events	19-11
SPMM Testrig	19-12
SPMM Output	19-13
SPMM Output - Postprocessing	19-14
Workshop 19: "SVC and SPMM Events in Adams Car"	



# **SECTION 0**

## **INTRODUCTION**

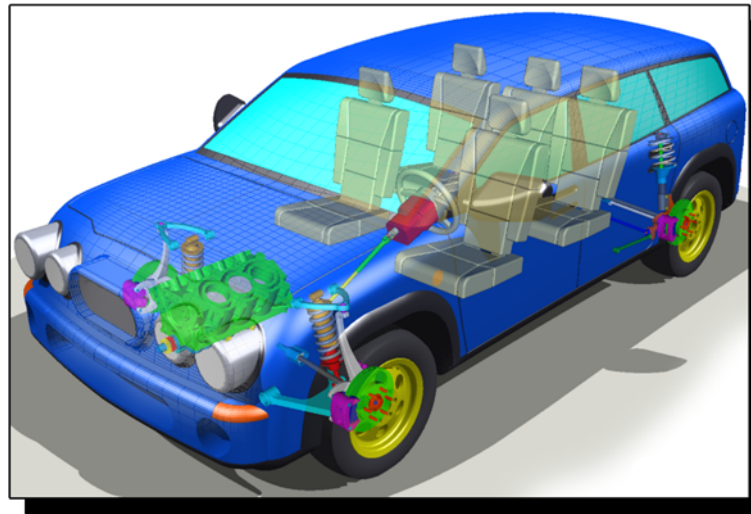
# WELCOME TO ADAMS/CAR TRAINING

- **What's in this section:**
  - Course Objectives
  - What is Adams?
  - What is Motion Simulation?
  - Adams Applications
  - Why Build an Adams Model?
  - SimCompanion: Technical Articles and Documentation
  - Organization of Workshops
  - Getting Help



# WELCOME TO ADAMS/CAR TRAINING

- **Welcome to Adams/Car training. Adams/Car is a specialized environment for modeling vehicles.**
- **It allows you to create virtual prototypes of vehicle subsystems and analyze the virtual prototypes much as you would analyze the physical prototypes.**



# WHAT IS ADAMS?

- **Automatic Dynamic Analysis of Mechanical Systems**
- **Development started in 1974 at the University of Michigan**
- **Mechanical Dynamics, Inc. started with Adams/Solver**
- **Euler-Lagrange method to create equations of motion**
- **Predictor-corrector methods to solve equations**
- **Integrated animation and plotting**
- **Powerful parametric, scripting and post-processing abilities.**



# WHAT IS MOTION SIMULATION?

- **Also called:**
  - Mechanical Systems Simulation (MSS)
  - Multi-Body Dynamics (MBD)
- **Equations of motion generated from the model**
- **CAD-like graphical interface or text file input**
- **Differential and algebraic equations integrated directly**
- **Results in system animation and plots of all kinematics (displacement, velocity, etc.) and dynamics (reactions, etc.)**

# ADAMS APPLICATIONS

- **The general purpose graphical user interface is Adams/View**
  - Model building, simulation submission, animation
- **Equations are built and integrated with Adams/Solver**
  - Solver is a standalone application, but typically used seamlessly from within Adams/View
- **Plotting and animation done with Adams/PostProcessor**
  - Tightly integrated with Adams/View
- **Industry verticals like Adams/Car**
  - A specialized environment for modeling vehicles using the templates based modeling approach.

# WHY BUILD AN ADAMS MODEL?

- **Verify Motion Performance (~60%)**
  - “Will it work?”
  - Example:
    - Will landing gear fully deploy in these conditions?
    - Will vehicle respond as desired in these conditions?
- **Compute Detailed Loads in a Mechanism (~30%)**
  - “Will it break?”
  - Example:
    - What are the cyclic loads on a wind turbine driveshaft?
    - What are the loads acting on different joints attached to body?
- **Examine Clearances in a Complex Mechanism (~10%)**
  - “Will it fit?”
  - Example:
    - Will launch vehicle fairing deploy without hitting the payload?
    - Will vehicle pass smoothly over the obstacle on the road?



# SIMCOMPANION

- One stop for full online support
- Find answers to your questions
- Subscribe to email notification
- Access to other support resources
  - Case Management Portal
  - Discussion Forums
  - Training Information

<https://simcompanion.mscsoftware.com>

**MSC Software**  
Simulating Reality. Delivering Certainty.

Register | Search | International | Blog | Contact Us

SERVICES TRAINING PRODUCTS ACADEMIA RESOURCES ABOUT US SUPPORT

Home | News | Tech Articles | Docs | Conference & Tech Papers | Examples | Issues | Product Updates | Webinars & Multimedia

English | Change

## Welcome to SimCompanion

The MSC Knowledge Base for Technical Articles, Documentation, Webinars & more

Recent Articles		Popular Articles	
ID	Title	ID	Title
KB8020723	Using the special FSAE database with Adams/Car 2012	KB8019304	Support Contact Information
KB8020844	Windows: Search Nastran TPL with findstr recursively	KB8008069	Errors in creation of EASY5 2010 ez5vars.bat file
KB8020834	Option to export EASY5 model as Adams External System Library is disabled	KB8006270	MSC Customer Entitlement ID (CEID) Description and Access Instructions
KB8020812	Sample C program to test Visual Studio C compiler configuration	KB8006390	Understanding MSC Web Account Access Levels
KB8020734	Patran Precision Control Options – null or no input of significant digits is allowed.	KB8019340	Installing Adams 2010 versions on Windows 7
KB8020090	Adams/Car for FSAE Main Directory	KB8018886	MSC's Solution Download Center (SDC) - Electronic Downloads
KB8019488	Using the special FSAE database with MD Adams/Car 2010	KB8008071	EASY5 2010 can't find Intel Fortran compiler
KB8008329	EASY5 2010.1.3 or later will not launch and gives error message stating that the application configuration is incorrect	KB8019501	Solution Download Center (SDC) - Adding New Users (admins only)
KB8020490	Accessing SimCompanion as an Academic User	KB8019842	How to use MSC's anonymous FTP Server
KB8020394	Forcing EASY5 to call ifortvars.bat	KB8008132	When trying to launch EASY5, 'which' command could not be found.

Recent Product News	
ID	Title
PN58	Patran 2010.2 Release Highlights
PN74	MSC Software Announces Release of Patran 2012.2.1
PN72	SimXpert 2012.0.1 Release Highlights
PN71	SimDesigner Suspension 2010.3

Welcome to SimCompanion

LOG IN | Password Help | First Time User

Get what you want from MSC more efficiently and effectively, either reactively through browsing and searching, or proactively through content subscriptions that you manage.

**Play Tour**

Click image above for a tour of SimCompanion.

**Additional Resources**

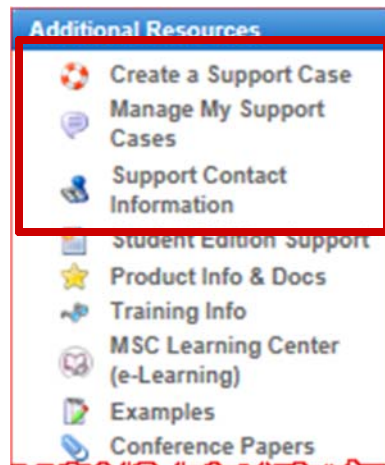
- Create a Support Case
- Manage My Support Cases
- Support Contact Information
- Product Info & Docs
- Training Info
- Conference Papers
- SimAcademy Webinar Series
- Technical Support Usage Guide
- SDC (Solution Download Center)
- FTP Instructions
- SimCompanion Help
- Give us Your Feedback

**Communities**

- Simulate More Blog
- Facebook
- Twitter
- VPD Community Forums
- YouTube
- Podcast

# SIMCOMPANION

- **Personalized Support via the following channels**
  - Web
    - Create a Support Case
    - Manage My Support Cases
  - Email
    - List of Addresses in Support Contact Information
  - Phone
    - List of Phone Numbers in Support Contact Information



<b>Web:</b>	
<a href="http://support.mssoftware.com/servicerequest">http://support.mssoftware.com/servicerequest</a>	
<b>Email:</b>	
English	
- SimCompanion Access	<a href="mailto:support@mssoftware.com">support@mssoftware.com</a>
- MSC Nastran	<a href="mailto:mscnastran.support@mssoftware.com">mscnastran.support@mssoftware.com</a>
- Adams	<a href="mailto:mscadams.support@mssoftware.com">mscadams.support@mssoftware.com</a>
- Patran	<a href="mailto:mscpatran.support@mssoftware.com">mscpatran.support@mssoftware.com</a>
- Actran	<a href="mailto:actran.support@mssoftware.com">actran.support@mssoftware.com</a>
- Dytran	<a href="mailto:mscdytran.support@mssoftware.com">mscdytran.support@mssoftware.com</a>
- Easy5	<a href="mailto:easy5.support@mssoftware.com">easy5.support@mssoftware.com</a>
- MSC Fatigue	<a href="mailto:mscfatigue.support@mssoftware.com">mscfatigue.support@mssoftware.com</a>
- Marc	<a href="mailto:mscmarc.support@mssoftware.com">mscmarc.support@mssoftware.com</a>
- MaterialCenter	<a href="mailto:materialcenter.support@mssoftware.com">materialcenter.support@mssoftware.com</a>
- Mvision	<a href="mailto:mscmvision.support@mssoftware.com">mscmvision.support@mssoftware.com</a>
- Sinda	<a href="mailto:mscsinda.support@mssoftware.com">mscsinda.support@mssoftware.com</a>
- SimDesigner	<a href="mailto:simdesigner.support@mssoftware.com">simdesigner.support@mssoftware.com</a>
- SimManager	<a href="mailto:simmanager.support@mssoftware.com">simmanager.support@mssoftware.com</a>
- Simufact (Americas)	<a href="mailto:support.americas@simufact.com">support.americas@simufact.com</a>
- Simufact (Outside Americas)	<a href="mailto:support@simufact.de">support@simufact.de</a>
- SimXpert	<a href="mailto:simxpert.support@mssoftware.com">simxpert.support@mssoftware.com</a>
- MSC Apex	<a href="mailto:mscapex.support@mssoftware.com">mscapex.support@mssoftware.com</a>
- MSC Licensing, Installation, and Configuration	<a href="mailto:msc_lic.support@mssoftware.com">msc_lic.support@mssoftware.com</a>
Chinese (Simplified)	<a href="mailto:support.cn@mssoftware.com">support.cn@mssoftware.com</a>
Chinese (Traditional)	<a href="mailto:support.tw@mssoftware.com">support.tw@mssoftware.com</a>
French	<a href="mailto:support.fr@mssoftware.com">support.fr@mssoftware.com</a>
German	<a href="mailto:support.de@mssoftware.com">support.de@mssoftware.com</a>
Italian	<a href="mailto:support.it@mssoftware.com">support.it@mssoftware.com</a>
Japanese	<a href="mailto:support.jp@mssoftware.com">support.jp@mssoftware.com</a>
Korean	<a href="mailto:support.kr@mssoftware.com">support.kr@mssoftware.com</a>
Portuguese	<a href="mailto:suporte.msobrasil@mssoftware.com">suporte.msobrasil@mssoftware.com</a>
Russian	<a href="mailto:support.ru@mssoftware.com">support.ru@mssoftware.com</a>
Spanish	<a href="mailto:support.es@mssoftware.com">support.es@mssoftware.com</a>
<b>Phone:</b>	
BeNeLux (Belgium, Netherlands, Luxembourg)	+31 (0)162 52400 (in Summa Innovation B.V.)
Brazil	0800-891-4346
China - Beijing	+86-10-82607000
China - Shanghai	+86-21-63326655
China - Chengdu	+86-28-86199275/6
China - Shenzhen	+86-755-23811695
Czech Republic	+420 54517 6106
DACH	+49 89 431 987 277
Denmark	+45 61 22 32 00
Finland	0800-9-14709
France	+33 5 34 60 44 20

# SIMCOMPANION

- **Product Info and Docs**
  - Access to all Product Documentation

### Additional Resources

- Create a Support Case
- Manage My Support Cases
- Support Contact Information
- Student Edition Support
- Product Info & Docs**
- Training Info
- MSC Learning Center (e-Learning)
- Examples
- Conference Papers
- SimAcademy Webinar Series
- Technical Support Usage Guide
- SDC (Solution Download Center)
- FTP Instructions
- SimCompanion Help
- Give us Your Feedback

### Docs

Product Information and Documentation

Docs ID: DOC9275  
Status: Published  
Published date: 06/25/2009  
Updated: 08/26/2016

**Description**

Please click on desired MSC Product icon, to find the summary of Product Information and Documentation for current and prior versions, such as:

- What's New
- Release Guides
- Hardware & Software Requirements
- Set Up Guides (Installation, Licensing, and Configuration)
- Other product-specific content...

**CAE Tools**

**Additional Resources**

- Create a Support Case
- Manage My Support Cases
- Support Contact Information
- Student Edition
- Product Info & Docs
- Training Info
- MSC Learning Center (e-Learning)
- Examples
- Conference Papers
- SimAcademy Webinar Series
- Technical Support Usage Guide
- SDC (Solution Download Center)
- FTP Instructions
- SimCompanion Help
- Give us Your Feedback

**Communities**

- Simulate More Blog
- Facebook
- Twitter
- VPO Community Forums

### Docs

Adams Product Information and Documentation

Docs ID: DOC9283  
Status: Published  
Published date: 06/27/2009  
Updated: 07/14/2017  
Reported In: Adams - Adams Docs Pattern

**Description**

Adams Product Information & Documentation

For viewing the Adams Gear AT documentation, Click [here](#).

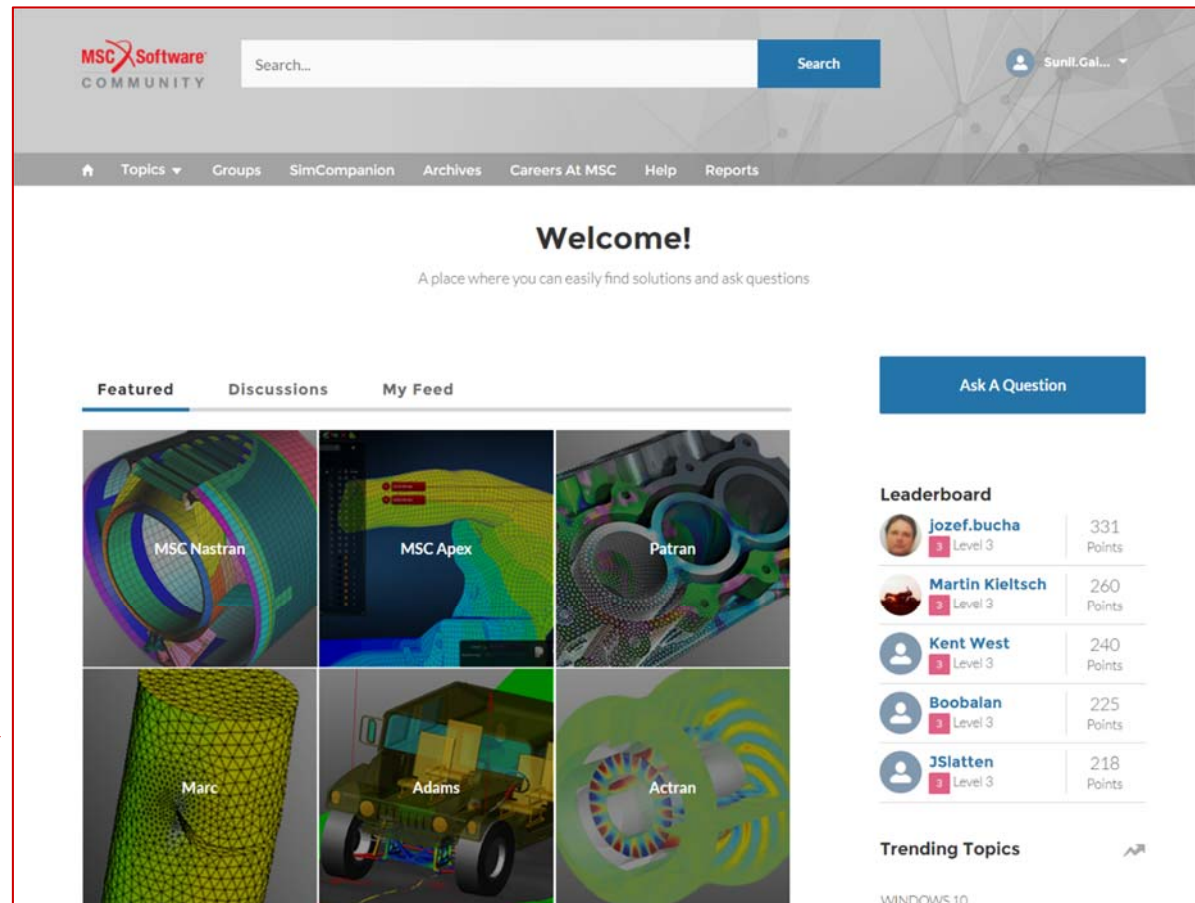
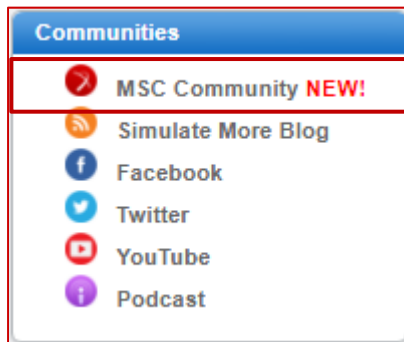
For MD Versions, Click [here](#).

	Version 2017.2	Version 2017.1	Version 2017	Version 2016	Version 2015.2	Version 2015.1.1	Version 2015.1	Version 2015	Version 2014 & 2014.1	Version 2013.2	Version 2013.1	Version 2013	Multikey (Version 2012.1.3)	Version 2012.1.2	Version 2011	Version 2010	Version 2008 v1
Release Notes (What's New, Changes, Issues, Feedback)	<a href="#">DOC11424</a>	<a href="#">DOC11283</a>	<a href="#">DOC11181</a>	<a href="#">DOC11046</a>	<a href="#">DOC11189</a>	<a href="#">DOC10971</a>	<a href="#">DOC10870</a>	<a href="#">DOC10804</a>	<a href="#">DOC10589</a>	<a href="#">DOC10383</a>	<a href="#">DOC10344</a>	<a href="#">DOC10338</a>	<a href="#">DOC10329</a>	<a href="#">DOC10183</a>	<a href="#">DOC10112</a>	<a href="#">DOC9972</a>	<a href="#">LSP &amp; Unlabeled</a>
What's New Summary	<a href="#">EX1403</a>	<a href="#">EX1400</a>	<a href="#">EX1437</a>	<a href="#">EX1422</a>			<a href="#">EX1405</a>	<a href="#">EX1407</a>	<a href="#">EX1406</a>	<a href="#">EX1405</a>	<a href="#">EX1404</a>	<a href="#">EX1403</a>					<a href="#">LSP &amp; Unlabeled</a>
Hardware & Software Requirements	<a href="#">See Release Guide (Chapter 2, page 11-12)</a>	<a href="#">See Release Guide (Chapter 2, page 11-12)</a>	<a href="#">See Release Guide (Chapter 2, page 11-12)</a>	<a href="#">See Release Guide (Chapter 2, page 11-12)</a>	<a href="#">See Release Guide (Chapter 2, page 11-12)</a>	<a href="#">See Release Guide (Chapter 2, page 11-12)</a>	<a href="#">See Release Guide (Chapter 2, page 11-12)</a>	<a href="#">See Release Guide (Chapter 2, page 11-12)</a>	<a href="#">See Release Guide (Chapter 2, page 11-12)</a>	<a href="#">See Release Guide (Chapter 2, page 11-12)</a>	<a href="#">See Release Guide (Chapter 2, page 11-12)</a>	<a href="#">See Release Guide (Chapter 2, page 11-12)</a>	<a href="#">See Release Guide (Chapter 2, page 11-12)</a>	<a href="#">See Release Guide (Chapter 2, page 11-12)</a>	<a href="#">See Release Guide (Chapter 2, page 11-12)</a>	<a href="#">See Release Guide (Chapter 2, page 11-12)</a>	<a href="#">LSP &amp; Unlabeled</a>
Supported Versions of Integration Products	<a href="#">See Release Guide (Chapter 4, page 70)</a>	<a href="#">See Release Guide (Chapter 4, page 70)</a>	<a href="#">See Release Guide (Chapter 4, page 70)</a>	<a href="#">See Release Guide (Chapter 4, page 70)</a>	<a href="#">See Release Guide (Chapter 4, page 70)</a>	<a href="#">See Release Guide (Chapter 4, page 70)</a>	<a href="#">See Release Guide (Chapter 4, page 70)</a>	<a href="#">See Release Guide (Chapter 4, page 70)</a>	<a href="#">See Release Guide (Chapter 4, page 70)</a>	<a href="#">See Release Guide (Chapter 4, page 70)</a>	<a href="#">See Release Guide (Chapter 4, page 70)</a>	<a href="#">See Release Guide (Chapter 4, page 70)</a>	<a href="#">See Release Guide (Chapter 4, page 70)</a>	<a href="#">See Release Guide (Chapter 4, page 70)</a>	<a href="#">See Release Guide (Chapter 4, page 70)</a>	<a href="#">See Release Guide (Chapter 4, page 70)</a>	<a href="#">LSP &amp; Unlabeled</a>
Release Guide	<a href="#">DOC11424</a>	<a href="#">DOC11283</a>	<a href="#">DOC11181</a>	<a href="#">DOC11046</a>	<a href="#">DOC11189</a>	<a href="#">DOC10971</a>	<a href="#">DOC10870</a>	<a href="#">DOC10804</a>	<a href="#">DOC10589</a>	<a href="#">DOC10383</a>	<a href="#">DOC10344</a>	<a href="#">DOC10338</a>	<a href="#">DOC10329</a>	<a href="#">DOC10183</a>	<a href="#">DOC10112</a>	<a href="#">DOC9972</a>	<a href="#">LSP &amp; Unlabeled</a>
Installation Guide	<a href="#">DOC11423</a>	<a href="#">DOC11284</a>	<a href="#">DOC11182</a>	<a href="#">DOC11046</a>	<a href="#">DOC11189</a>	<a href="#">DOC10972</a>	<a href="#">DOC10871</a>	<a href="#">DOC10807</a>	<a href="#">DOC10588</a>	<a href="#">DOC10382</a>	<a href="#">DOC10343</a>	<a href="#">DOC10338</a>	<a href="#">DOC10329</a>	<a href="#">DOC10183</a>	<a href="#">DOC10113</a>	<a href="#">DOC9974</a>	<a href="#">LSP &amp; Unlabeled</a>
Configuring Adams	<a href="#">DOC11428</a>	<a href="#">DOC11285</a>	<a href="#">DOC11183</a>	<a href="#">DOC11046</a>													<a href="#">LSP &amp; Unlabeled</a>
About Adams	<a href="#">DOC11427</a>	<a href="#">DOC11286</a>	<a href="#">DOC11184</a>	<a href="#">DOC11046</a>													<a href="#">LSP &amp; Unlabeled</a>
<b>Getting Started Guides</b>																	
Getting Started Using Adams Car	<a href="#">DOC11438</a>	<a href="#">DOC11287</a>	<a href="#">DOC11188</a>	<a href="#">DOC11044</a>			<a href="#">DOC10981</a>	<a href="#">DOC10980</a>	<a href="#">DOC10821</a>			<a href="#">DOC10384</a>		<a href="#">DOC10386</a>	<a href="#">DOC10377</a>	<a href="#">DOC9976</a>	<a href="#">LSP &amp; Unlabeled</a>
Getting Started Using Adams Car Truck	<a href="#">DOC11439</a>	<a href="#">DOC11286</a>	<a href="#">DOC11188</a>	<a href="#">DOC11044</a>			<a href="#">DOC10980</a>	<a href="#">DOC10979</a>	<a href="#">DOC10822</a>			<a href="#">DOC10385</a>		<a href="#">DOC10386</a>	<a href="#">DOC10378</a>	<a href="#">DOC9977</a>	<a href="#">LSP &amp; Unlabeled</a>
Getting Started Using Adams Car Truck	<a href="#">DOC11410</a>	<a href="#">DOC11289</a>	<a href="#">DOC11187</a>	<a href="#">DOC11046</a>			<a href="#">DOC10980</a>	<a href="#">DOC10912</a>	<a href="#">DOC10823</a>			<a href="#">DOC10386</a>		<a href="#">DOC10384</a>	<a href="#">DOC10379</a>	<a href="#">DOC9978</a>	<a href="#">LSP &amp; Unlabeled</a>
Getting Started Using Adams Chassis	<a href="#">DOC11411</a>	<a href="#">DOC11289</a>	<a href="#">DOC11188</a>	<a href="#">DOC11047</a>			<a href="#">DOC10980</a>	<a href="#">DOC10914</a>	<a href="#">DOC10824</a>			<a href="#">DOC10387</a>		<a href="#">DOC10380</a>	<a href="#">DOC10380</a>	<a href="#">DOC9979</a>	<a href="#">LSP &amp; Unlabeled</a>



# SIMCOMPANION

- **Access to Communities**
  - VPD Community Discussion Forums
  - Subscribe to discussion communities of interest



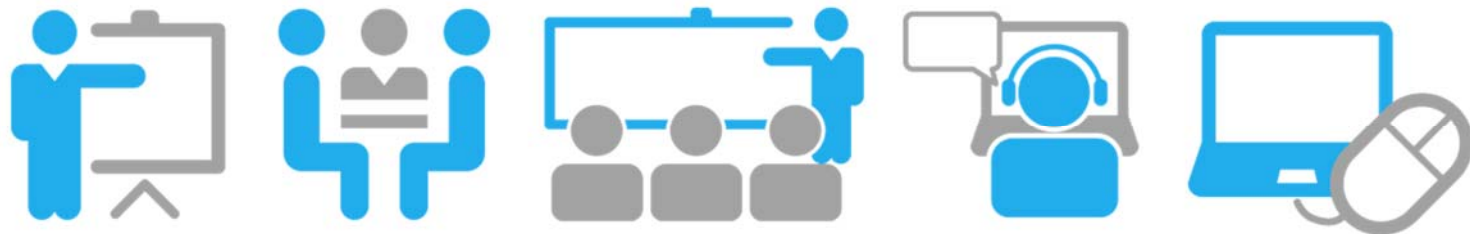
# MSC e-LEARNING



- **Features:**
  - Online Interactive Courses with hands on workshops
  - Certification Exams
  - Help and Support with Community
- **With MSC e-Learning subscription you get access to all courses for full year.**
- **Register for free trial at**
  - <http://www.mscsoftware.com/msc-training#self>
- **Send an email to [learning@mscsoftware.com](mailto:learning@mscsoftware.com) to request your subscription**


# ORGANIZATION OF WORKSHOPS

- **The workshops are majorly divided into four parts:**
  - Introduction to Adams/Car
  - Focus on Standard Interface
  - Focus on Template Builder
  - Focus on Plugins to Adams/Car
- **The earlier workshops provide you with more step-by-step procedures and guidance, while the later ones provide you with less.**



# GETTING HELP – ONLINE HELP

- **Online Help**

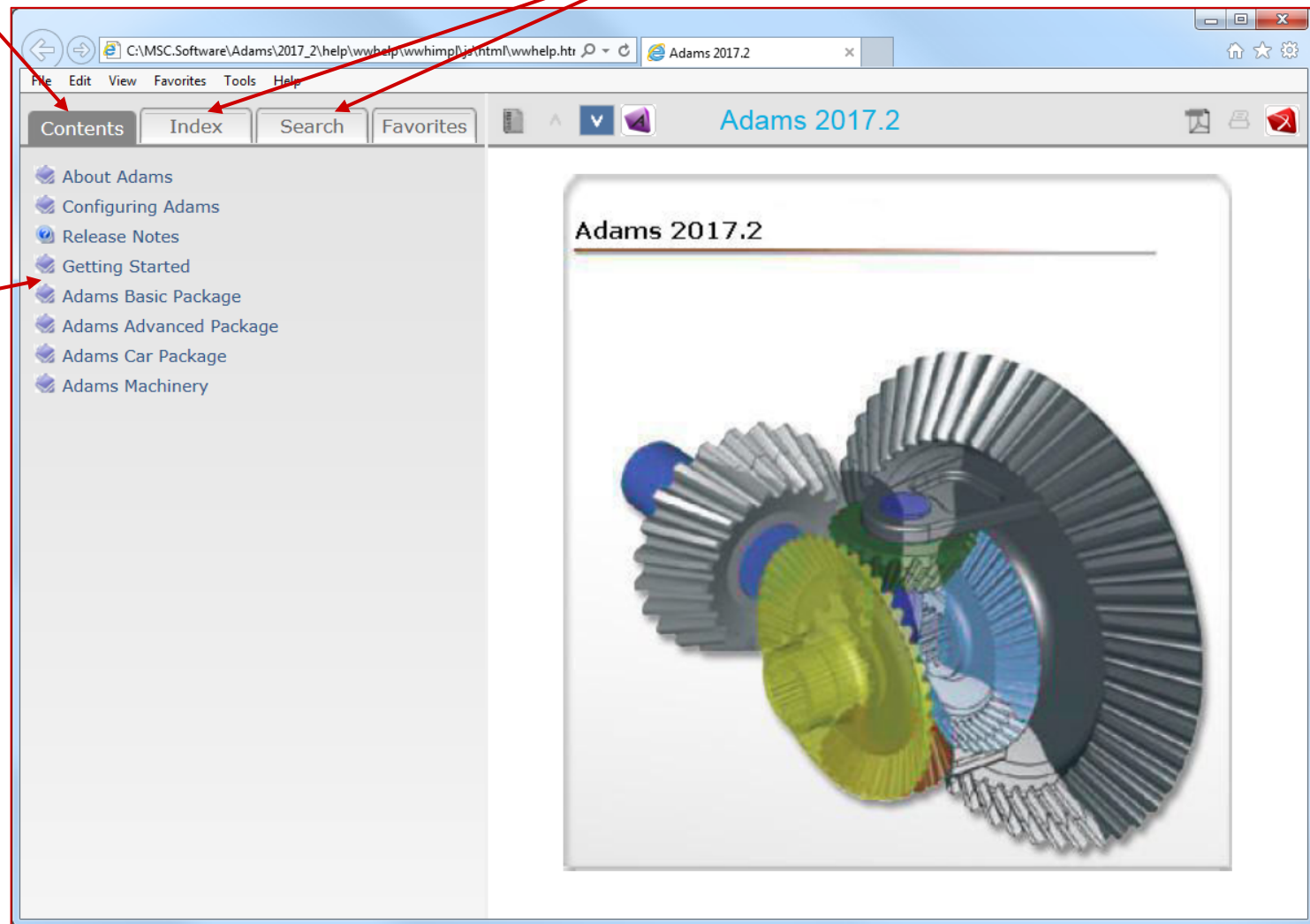
- To access the online help, do either of the following:
  - From the **main menu** bar, right-click on icon  and select **Adams/Car Help** to display the home page for the Adams/Car online help.
  - On Windows, go to **Start → All Programs → MSC.Software → Adams 2017.2 → Adams Online Help**
  - While working in any Adams/Car dialog box, press **F1 to display online help** specific to that dialog box.
- Once the online help is displayed, you can browse the table of contents, use the index, or search for keywords.

# GETTING HELP – ONLINE HELP

Contents of  
selected tab

Index/search entire  
Adams help

Table of  
contents

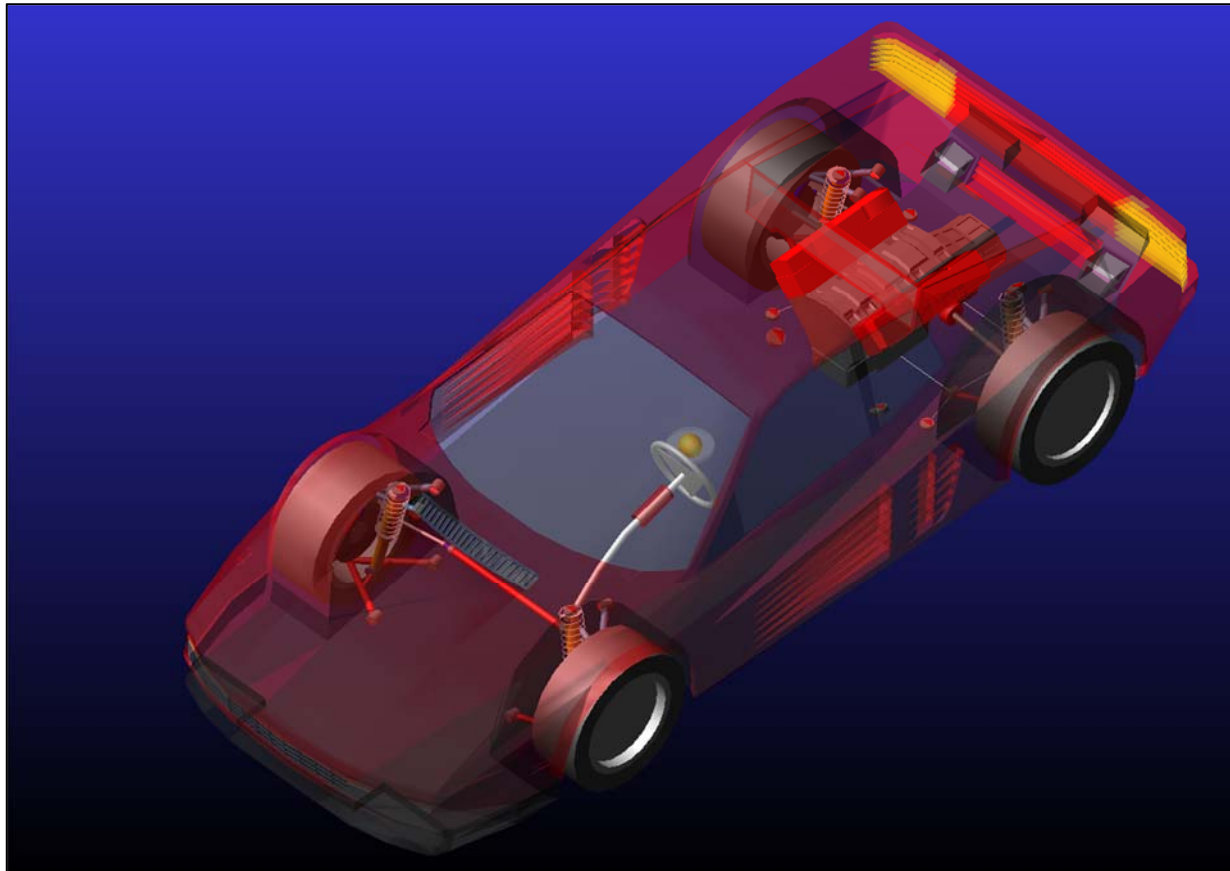


# GETTING HELP – CONSULTING

- **MSC.Software provides comprehensive engineering consulting services to help you realize the benefits of Virtual Product Development quickly and confidently.**
  - Staff Augmentation
  - Technology Transfer
  - Customization & Process Automation
  - Methods Development
  - Toolkit Solutions
  - Simulation Data and Process Management
  - On-site Support
- **For more information on MSC Consulting Services, go to:**  
<http://www.mscsoftware.com/page/expert-demand>

# SECTION 1

## INTRODUCING ADAMS/CAR



# INTRODUCING ADAMS/CAR

- **This module discusses the advantages of using Adams/Car, as well as the organization of the basic files.**

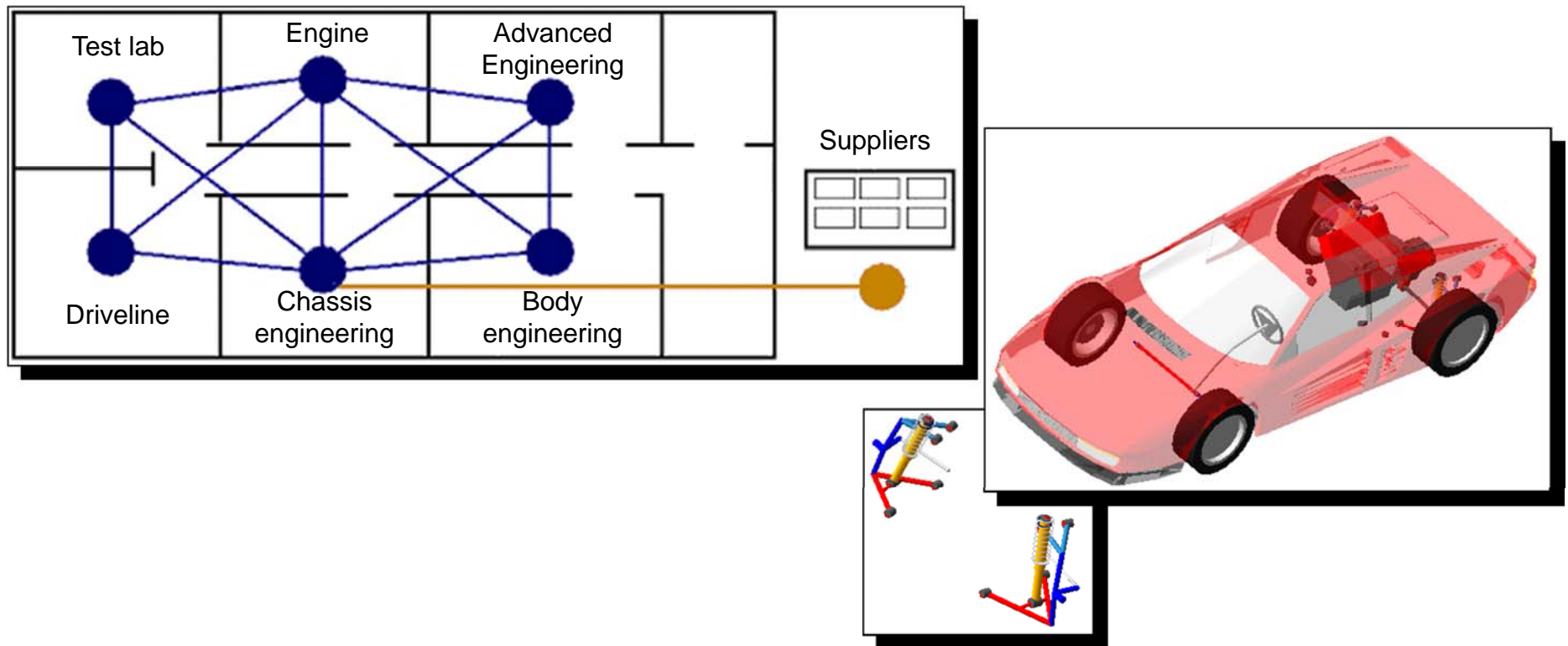


# INTRODUCING ADAMS/CAR

- **What's in this section:**
  - Motivation for Using Adams/Car
  - User Modes
  - Database Structure - A Directory Hierarchy
  - Saving Files: Working Directory versus Database
  - Configuration Files

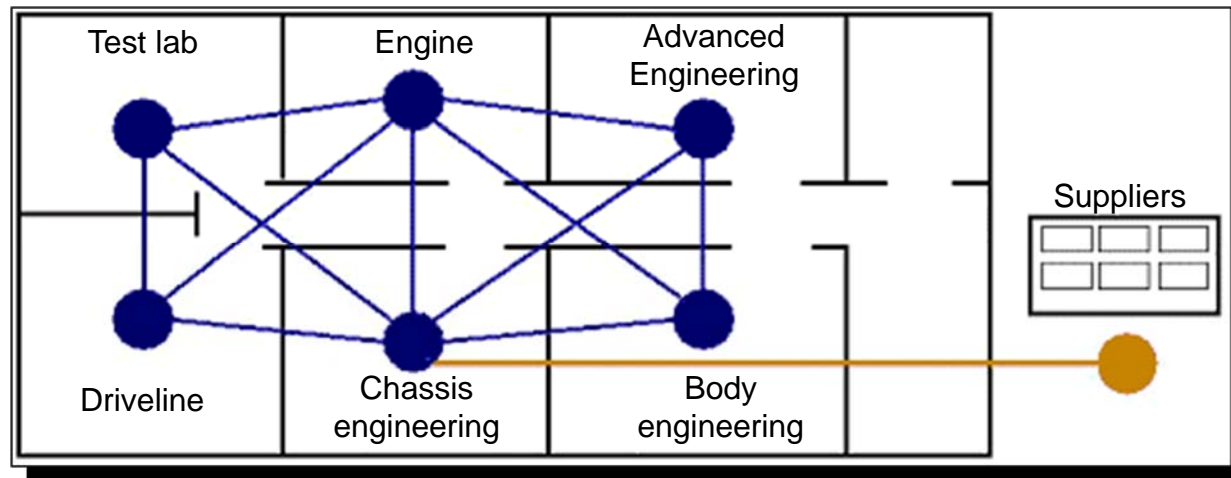
# MOTIVATION FOR USING ADAMS/CAR

- Bridges departments by sharing models and data
- Facilitates quick subsystem changes
- Templates



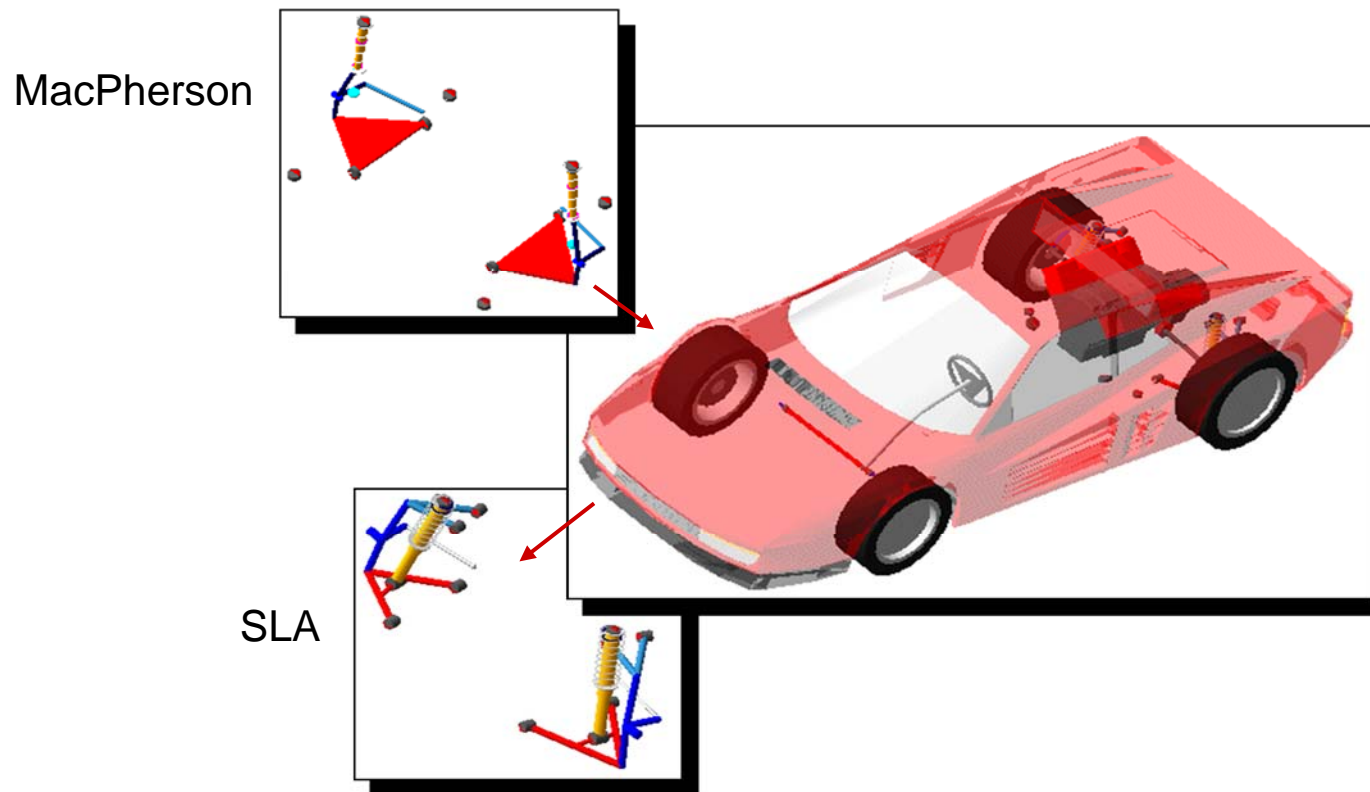
# MOTIVATION FOR USING ADAMS/CAR

- **Bridges departments by sharing models and data**
  - Different departments can work with the same database, which minimizes data loss.



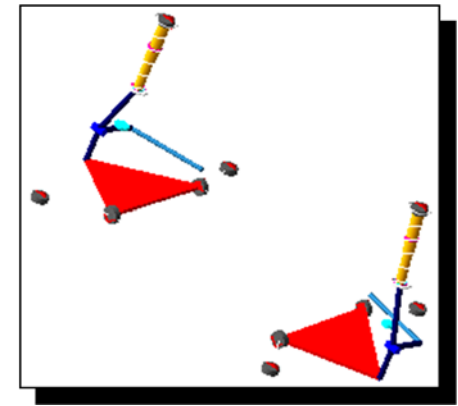
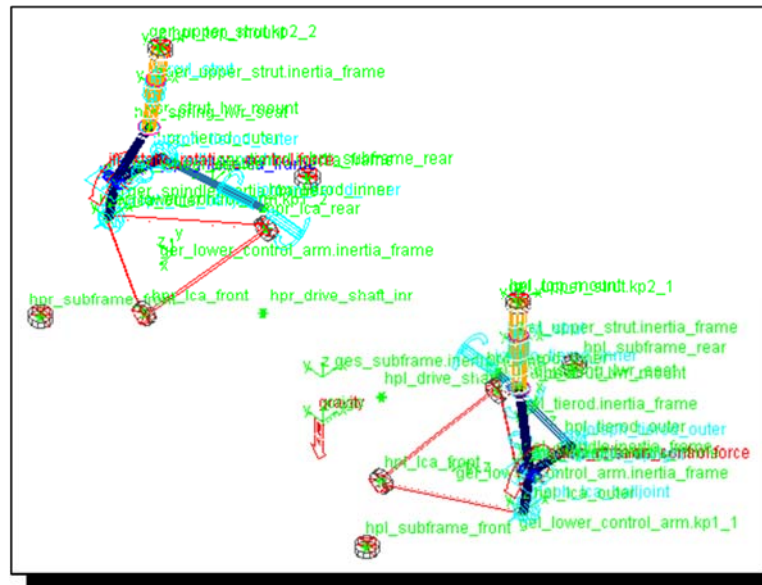
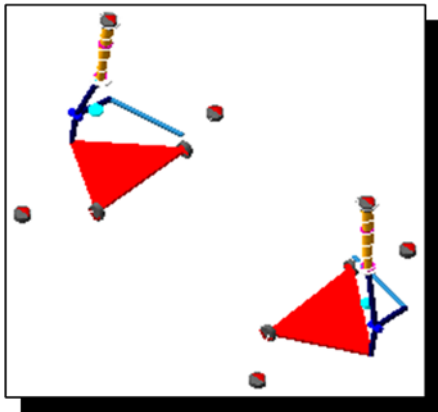
# MOTIVATION FOR USING ADAMS/CAR

- **Facilitates quick subsystem changes**
  - You can easily replace one subsystem without changing any other part of the vehicle.



# MOTIVATION FOR USING ADAMS/CAR

- **Templates**
  - Allow you to tailor one system for multiple vehicles.



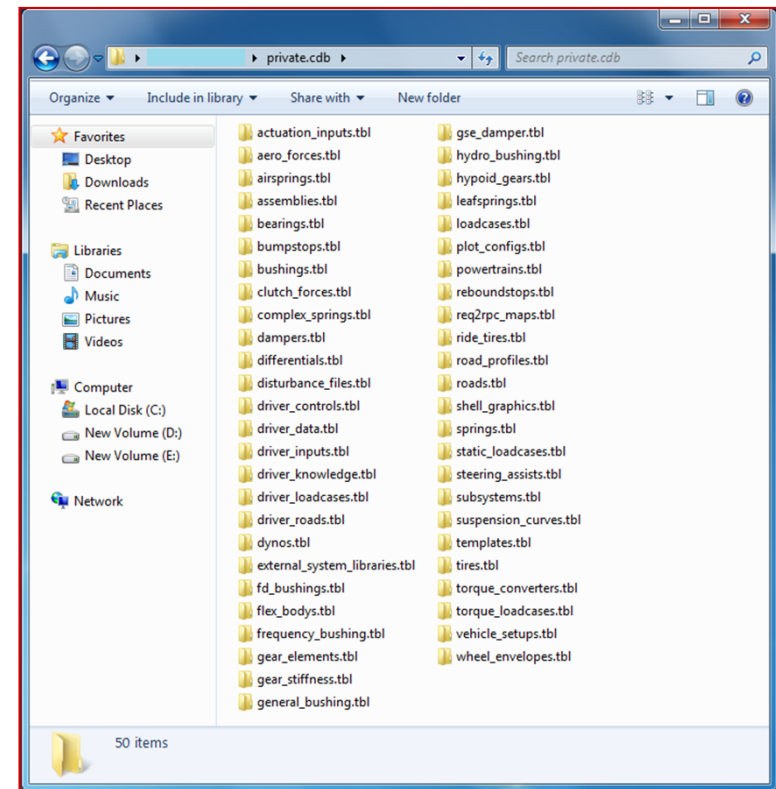
# USER MODES

- Within the Adams/Car configuration file (acar.cfg and .acar.cfg), the particular application of Adams/Car is specified as either standard user mode or expert user mode.
- **Standard user (Standard Interface only)**
  - Specifically for designers and testing engineers
  - Use libraries from the Adams/Car database to easily create vehicle (sub)assemblies
  - Simulation environment tailored to automotive standards
- **Expert user (Template Builder and Standard Interface)**
  - Allows creation of building-block of Adams/Car, templates, with access to Template Builder
  - For experienced Adams users
  - Access to all Adams modeling entities

```
1 !*****!
2 !***      Car Database Configuration Info      ***!
3 !***      ***!
4 !***      Do NOT edit this file, for configuration changes edit the ***!
5 !***      private configuration file found in the $HOME directory ***!
6 !***      See the ADAMS documentation for more details ***!
7 !*****!
8 ! Desired user mode (standard/template_builder)
9 ENVIRONMENT MDI_ACAR_USERMODE standard
10 ENVIRONMENT MDI_ACAR_MODEPROMPT yes
11 !
```

# DATABASE STRUCTURE - A DIRECTORY HIERARCHY

- A database is a collection of tables (directories) stored on the hard drive.
- The top directory, which has the extension .cdb, stores a number of tables (directories).
- Three types of databases:
  - Shared - Common to all users, provided by MSC.Software with example files
  - Private - User workspace (created by Adams/Car in your \$HOME directory)
  - User - User/site specific



# DATABASE STRUCTURE - A DIRECTORY HIERARCHY

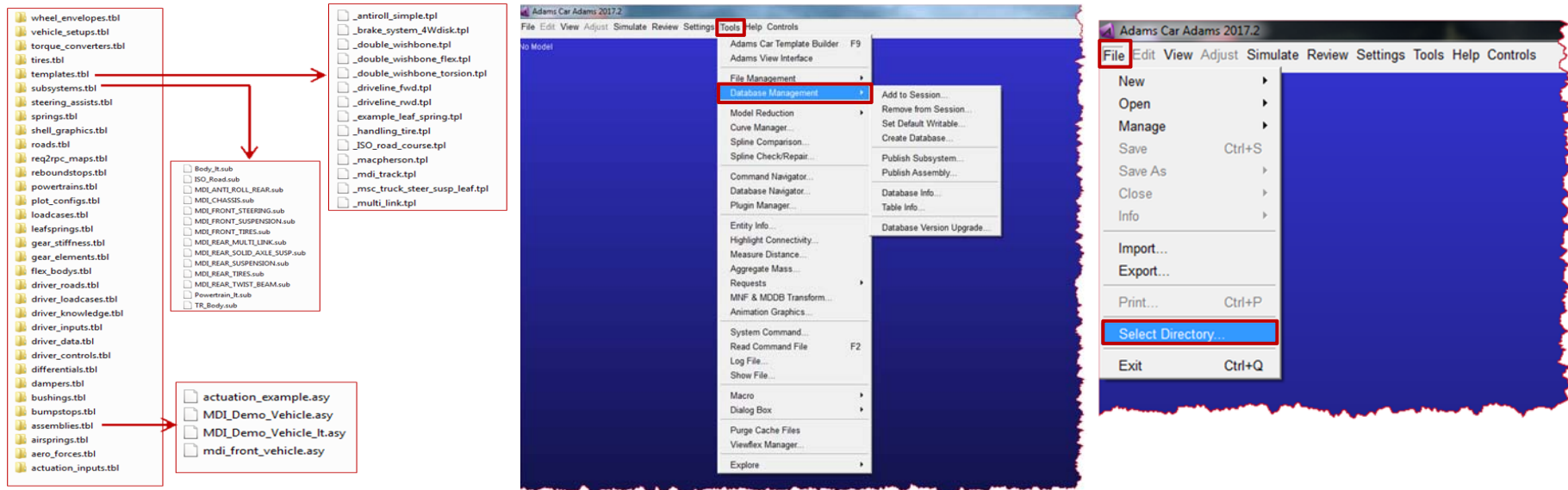
- The databases are defined in private .acar.cfg or common acar.cfg
- No limitations on number of databases
- Each project should have a separate database
- You can only save to one database at a time
- For more information about managing Adams/Car databases, see the Adams/Car online help.





# SAVING FILES: WORKING DIRECTORY VERSUS DATABASE

- Database stores your modeling components. You can set up the default writable database using the database management tools (**Tools → Database Management**).
- When solving an event Adams/Car saves the .adm, .acf, and results to your working directory (**File → Select Directory**).



# CONFIGURATION FILES

- **Configuration files (acar.cfg and .acar.cfg) contain information that Adams/Car reads during startup to correctly initialize the session:**
  - User mode (expert versus standard)
  - Personal databases and tables
  - Default property files
  - Default writable database
  - Database search order
  - Orientation of global reference frame
  - Other preferences

# SHARED VERSUS PRIVATE CONFIGURATION FILES

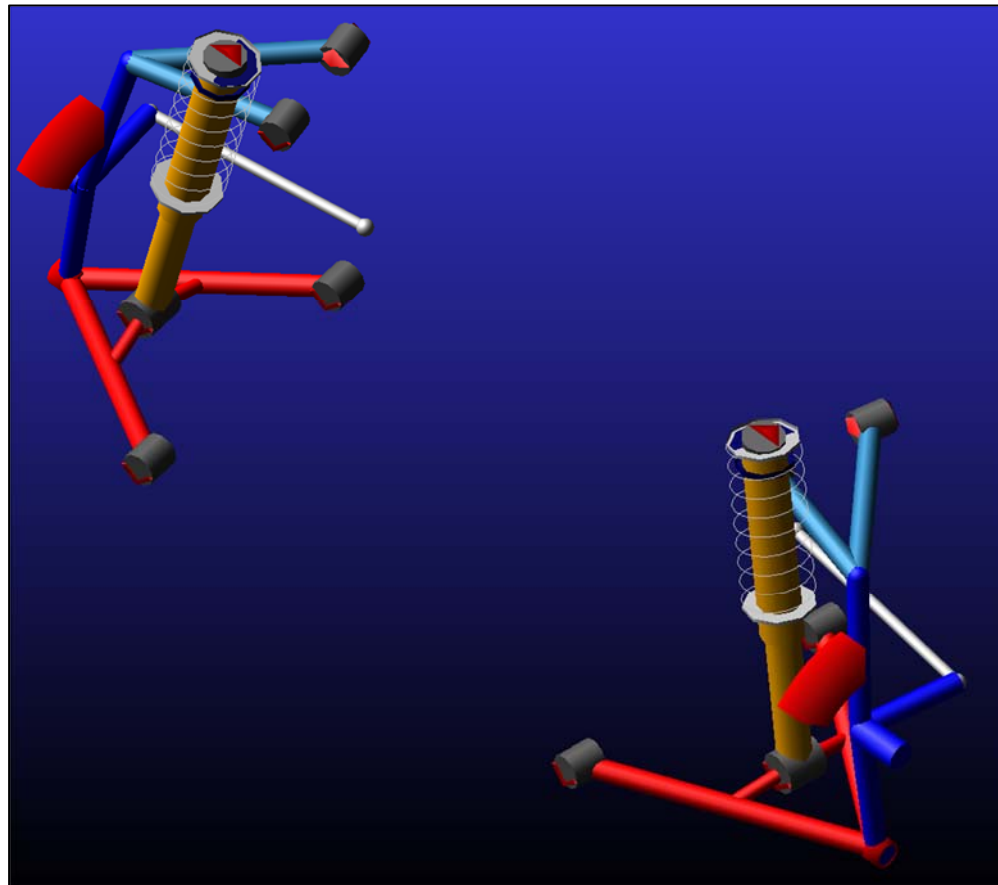
- **Adams/Car uses two configuration files:**
  - Shared configuration file (acar.cfg) –
    - Is a common file that sets up default parameters (read first).
    - Can be a specially created site-specific file.
    - This file is located in the <adams\_install>/acar directory (where <adams\_install> is the path to your Adams installation directory).
  - Private configuration file (.acar.cfg - note the period at the front) –
    - Is a private file that also sets up parameters, but takes precedence over the common acar.cfg file (read second).
    - The private configuration file is located in the HOME directory (environment variable).
- **Having both configuration files allow users to work from default parameters to customized parameters for their own workstations with the private .acar.cfg file.**

# EXERCISE

- **Perform Workshop 1, “OPEN AND RUN AN ASSEMBLY”.**

# SECTION 2

## BASIC CONCEPTS



# BASIC CONCEPTS

- **In this section, you will learn how to create models in Adams/Car. This module describes the relationships and differences between templates, subsystems and assemblies.**

# BASIC CONCEPTS

- **What's in this section:**
  - A Review of Basic Adams Terminology
  - Data Hierarchy
  - Test Rig
  - Major and Minor Roles
  - Naming Convention

# A REVIEW OF BASIC ADAMS TERMINOLOGY

- **Adams/Solver**
  - The solution engine.
- **Adams/Solver dataset (\*.adm)**
  - The ASCII file submitted to Adams/Solver, which contains statements and functions that define the model.
- **Adams/Solver command (\*.acf)**
  - An ASCII file that contains commands to control how Adams/Solver runs the model.



# A REVIEW OF BASIC ADAMS TERMINOLOGY

- **Adams/Solver output files**

- Graphics (\*.gra) - Information about how graphics work.
- Request (\*.req) - Contains output for user-specified results. This file can not be produced by Adams/Car by default.
- Results (\*.res) - Contains state-like results for every entity. This file can get very big and you can change its contents. Required to animate flexible bodies.
- Message (\*.msg) - Information about the Adams/Solver simulations. Useful for debugging with DEBUG/EPRINT.
- Output (\*.out) - Output including initial conditions and request, and content can depend on output specifications. It is usually tabular and human readable (for example, tables for output from LINEAR command).

# DATA HIERARCHY

- **Three levels of files build up a vehicle model (full or half vehicle):**

- Template

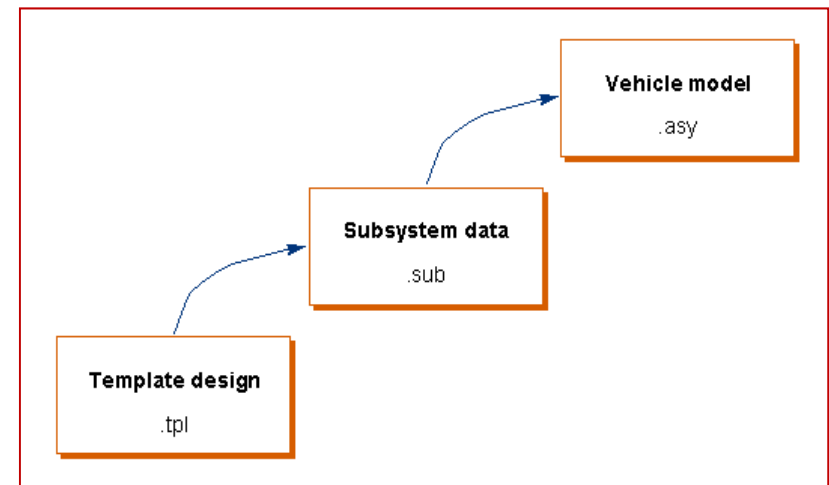
- Templates are parametric models, built by expert users within the Template Builder. Templates define the default geometric data and topology of models.

- Subsystem

- You only use subsystems in the Standard Interface. Subsystems are based on templates and allow standard users to change the parametric data of the template as well as the definition of some of the components.

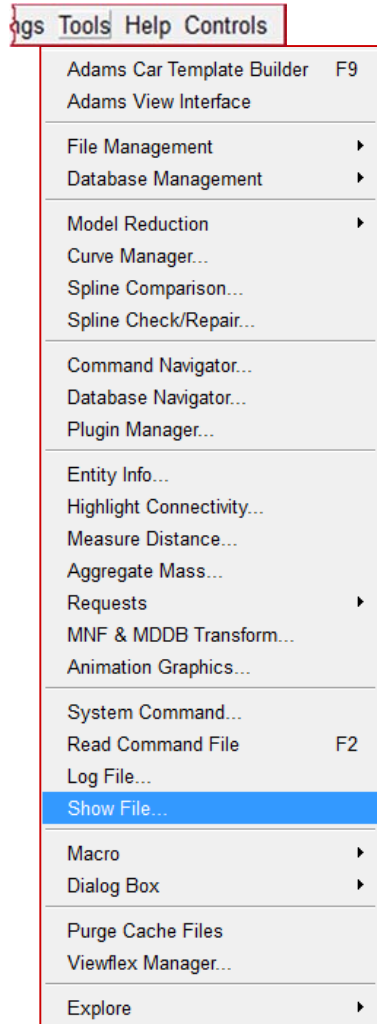
- Assembly

- Assemblies represent a collection of subsystems, along with a test rig, which when assembled form a system that you can analyse using Adams/Solver.



# DATA HIERARCHY

- You can see the hierarchy of an assembly by selecting **Tools → Show File (mdi\_front\_vehicle.asy):**



```
$-----MDI_HEADER
[MDI_HEADER]
FILE_TYPE      = 'asy'
FILE_VERSION    = 1.0
FILE_FORMAT     = 'ASCII'
HEADER_SIZE     = 9
(COMMENTS)
{comment_string}
'Adams/Car suspension assembly'
$-----ASSEMBLY_HEADER
[ASSEMBLY_HEADER]
ASSEMBLY_CLASS  = 'suspension'
TIMESTAMP       = '2009/09/25,18:32:00'
HEADER_SIZE     = 5
$-----PLUGINS
[PLUGINS]
PLUGIN_LIST     = 'acar'
HEADER_SIZE     = 4
$-----UNITS
[UNITS]
LENGTH         = 'mm'
FORCE           = 'newton'
ANGLE           = 'deg'
MASS            = 'kg'
TIME            = 'sec'
$-----SUBSYSTEM
[SUBSYSTEM]
$ Subsystem information:
$   Major Role : suspension
$   Minor Role : front
$   Template   : _macpherson
$
$   USAGE      = '<acar_shared>/subsystems.tbl/MDI_FRONT_SUSPENSION.sub'
$-----SUBSYSTEM
[SUBSYSTEM]
$ Subsystem information:
$   Major Role : steering
$   Minor Role : front
$   Template   : _rack_pinion_steering
$
$   USAGE      = '<acar_shared>/subsystems.tbl/MDI_FRONT_STEERING.sub'
```

# TEST RIG

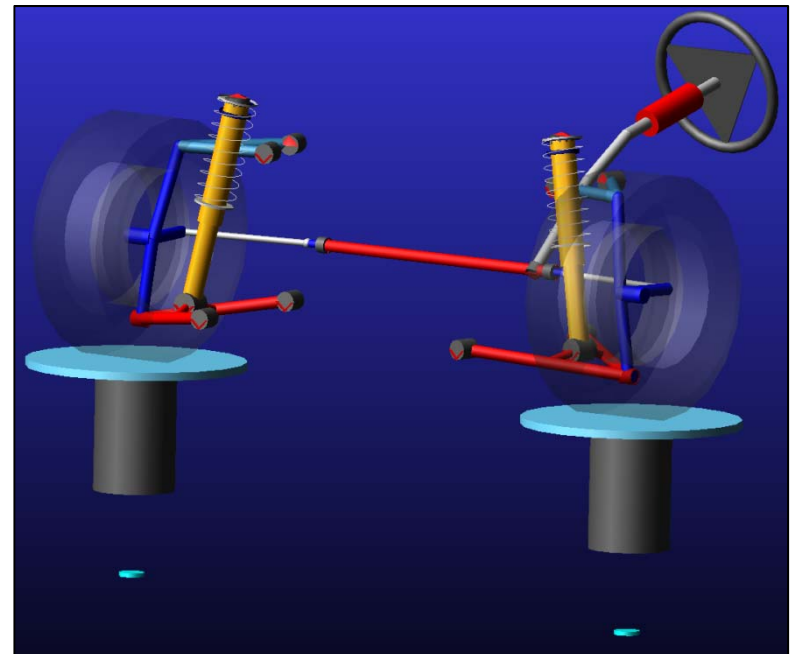
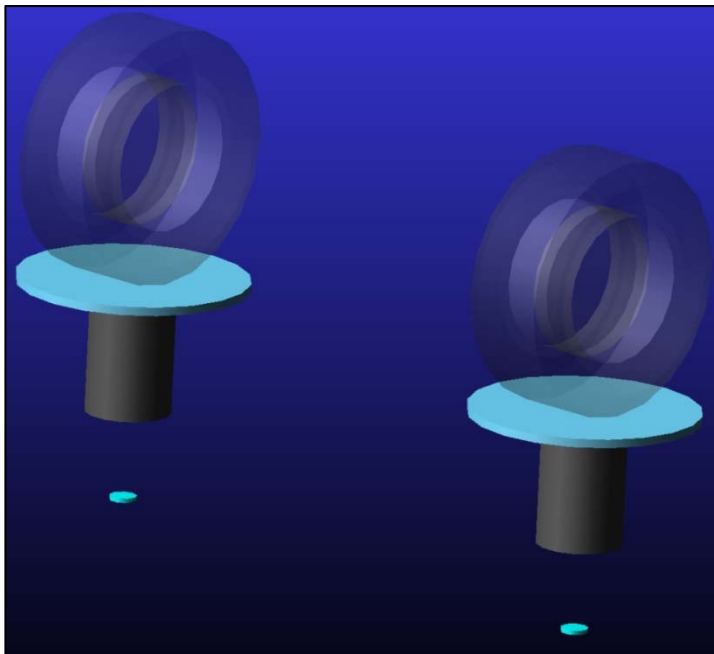
- **A test rig in Adams/Car is the part of the model that imposes motion on the vehicle. Depending on the model and event, different test rigs must be used.**
- **A test rig is a special subsystem that is connected to all of the other subsystems that make up your model, forming an assembly.**

# TEST RIG

- **Suspension Test Rig**

- The figure on the left shows the suspension test rig alone. With the suspension subsystems, it would look like the figure on the right (an assembly).

**Suspension test rig alone**

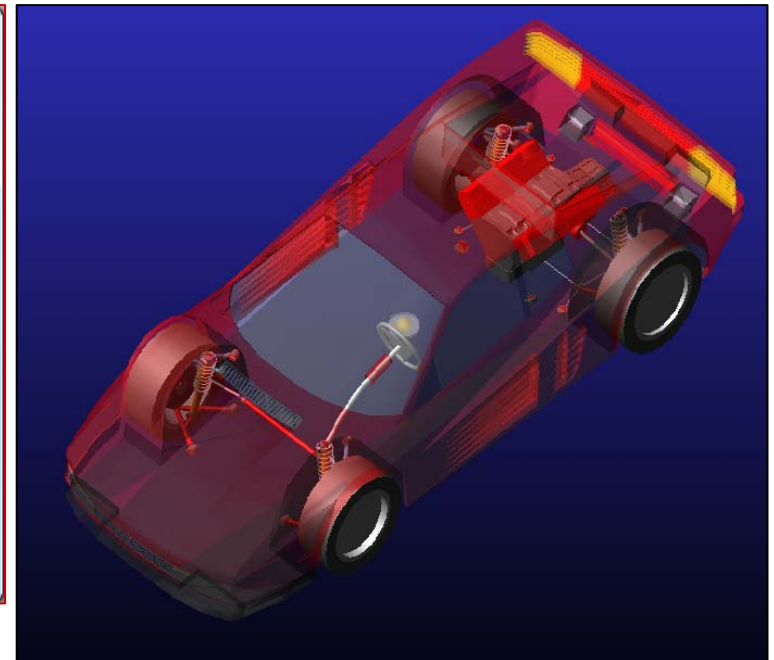
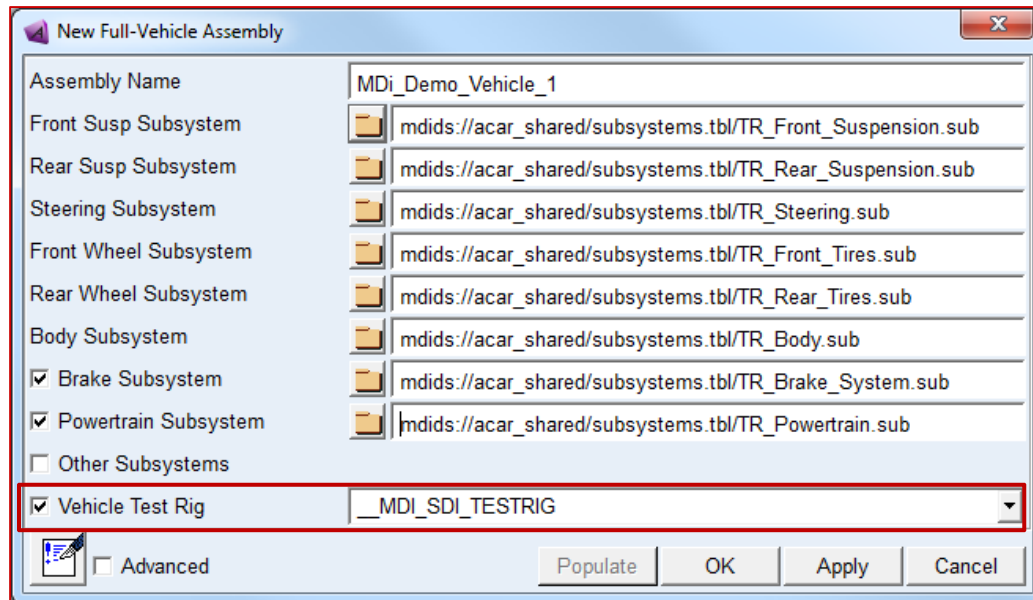


# TEST RIG

- **Full Vehicle Test Rig**

- The figure below shows the full vehicle assembled with full vehicle test rig.

## Full vehicle test rig assembled with required subsystems



# MAJOR AND MINOR ROLES

- **Adams/Car uses major and minor roles to create a valid assembly.**
- **Major and minor roles define the location of the subsystem within the assembly.**
- **Every template (and therefore the subsystem that is created from that template) has a defined major role: *suspension, steering, body, anti-roll bar, wheel*, and so on.**
- **When a subsystem is created, the standard user defines the subsystem minor role: *front, rear, trailer*, or *any*. This enables the same suspension template to be used for both a front and rear suspension.**

# MAJOR AND MINOR ROLES

- To add more major or minor roles, use an acarBS.cmd in your working directory with the following commands:

## Major roles

```
!-- Add a new major role ----  
variable modify variable=.ACAR.variables.major_roles &  
string_value=(eval(.ACAR.variables.major_roles)), "subframe"
```

## Minor roles

```
!-- Add a new minor role ----  
variable modify variable=.ACAR.variables.minor_roles &  
string_value=(eval(.ACAR.variables.minor_roles)), "rear2"
```



# NAMING CONVENTION

- **All Adams/Car entities are named after a naming convention. The first three letters of a given entity identify the type and the symmetry rule**
- **Examples:**
  - gel\_arm: General\_Part\_Left\_....
  - hps\_lcs\_front: Hard\_Point\_Single\_...
  - bkl\_mount: Bushing\_Kinematic\_Left\_...
  - nsr\_main\_spring: Non-linear\_Spring\_Right\_...
  - pvs\_toe\_angle: ParameterVariable\_Visible\_Single\_...



**For a complete listing, see the Adams/Car online help:  
Working with Components → Introducing the components →  
About the Naming Convention.**

# EXERCISE

- **Perform Workshop 2, “TEMPLATE VERSUS SUBSYSTEMS”.**

# SECTION 3

## CREATING AND ADJUSTING SUBSYSTEMS



# CREATING AND ADJUSTING SUBSYSTEMS

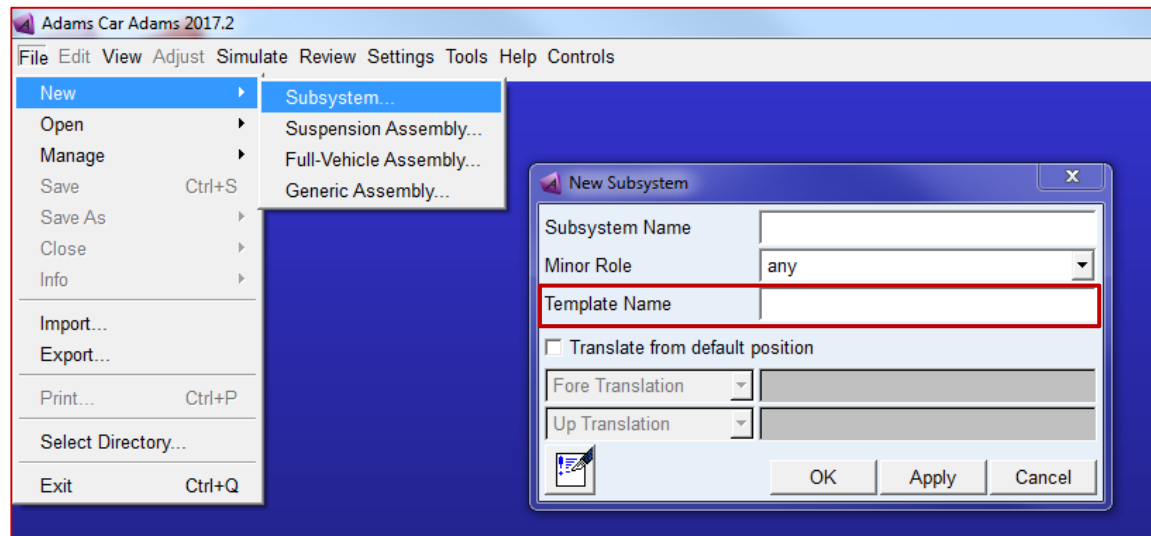
- **In this section, you will learn how to create a subsystem from a template, as well as learn which parameters you can adjust in the subsystem.**

# CREATING AND ADJUSTING SUBSYSTEMS

- **What's in this section:**
  - Creating Subsystems
  - Modifying Subsystems
    - Adjusting Hardpoints
    - Adjusting Parameter Variables
    - Adjusting Mass Properties
    - Adjusting Springs and Dampers
    - Property Files
      - Introduction
      - Curve Manager
      - Property File Editor
      - Saving to default writable
      - Converting TeimOrbit Files to XML
    - Replacing Instance Definition
    - More Subsystem Adjustments

# CREATING SUBSYSTEMS

- To create a new subsystem, an existing template must be available.
- Because you can only create subsystems within Standard Interface, make sure you're in Standard Interface.
- From the File menu, point to New and then select Subsystem.



- In the New Subsystem dialog box, fill in the following text boxes:
  - Subsystem Name
  - Minor Role
  - Template Name
  - Translation values (optional; lateral shifting cannot be done)

# MODIFYING SUBSYSTEMS

- **In this section, we'll cover the following ways to modify a subsystem:**
  - Adjusting Hardpoints
  - Adjusting Parameter Variables
  - Adjusting Mass Properties
  - Adjusting Springs and Dampers
  - Property Files
  - Replacing Instance Definition
  - More Subsystem Adjustments

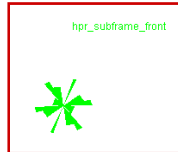
# ADJUSTING HARDPOINTS

- **Within a subsystem, you can move hardpoints from their default values defined in the template. Hardpoints define all key locations in your model. For more information on hardpoints, see Creating Hardpoints, in Section 12.**
- **In Standard Interface, from the Adjust menu, select Hardpoint. You have three options:**
  - **Modify** - Displays a dialog box to select one hardpoint and modify its location.
  - **Table** - Displays a table with all the hardpoints in that subsystem. You can modify the location of any hardpoint in the table.
  - **Info** - Displays a dialog box to select entity type and subsystem. This is already preselected to entity type of hardpoint and to the current subsystem. It will give you information about every hardpoint in the subsystem.

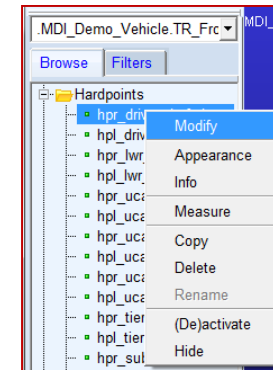


# ADJUSTING HARDPOINTS

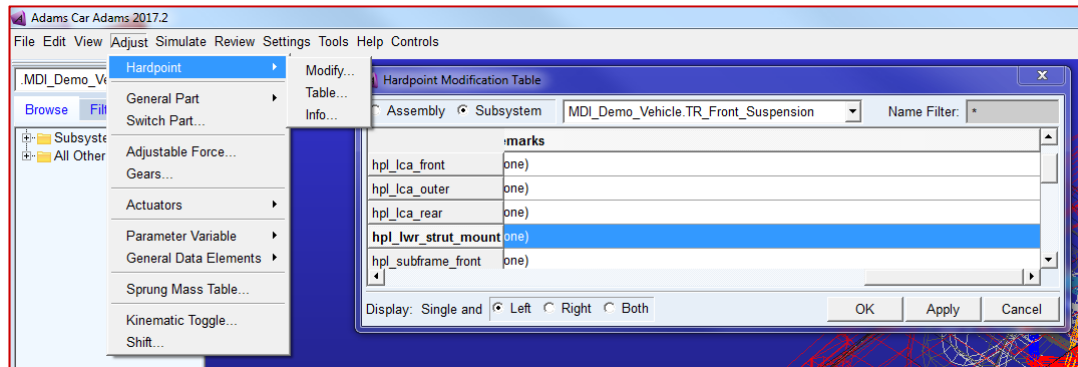
- **Modify the control arm chassis attachment hardpoint**



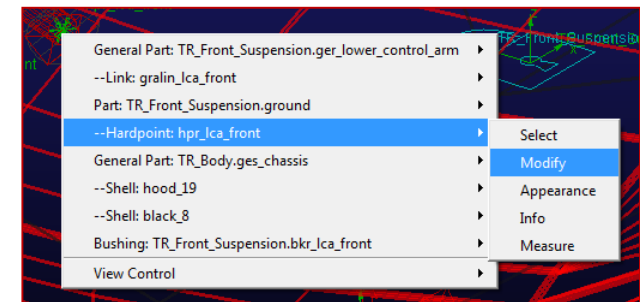
**Hardpoint Graphic**



**Using Model Browser**



**Using Adjust Menu**



**Using GUI**

# ADJUSTING PARAMETER VARIABLES

- **Within a subsystem, you can change the value of parameter variables created in Template Builder.**
- **A parameter variable is simply a variable that is used to store key information in the template.**
- **For example, in the templates, parameter variables often store the toe and camber angles for a suspension or the orientation of the spin axis. Note that parameter variables can also store text.**

# ADJUSTING PARAMETER VARIABLES

- **To modify parameter variables, from the Adjust menu, select Parameter Variable. You have two options:**
  - **Modify** - Displays a dialog box to select one parameter variable and modify its value.
  - **Table** - Displays a table with all parameter variables in that subsystem and you can modify the value of any parameter variables in the table.
- **Check alternately, from the Pull down menu at top of Model Browser select Subsystem to which the Parameter Variable you want to modify belongs. Point to Parameters, right click the Parameter variable you want to modify and select Modify.**

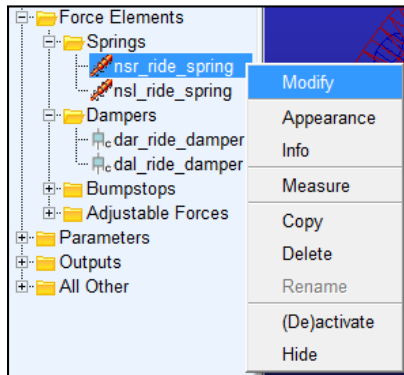
# ADJUSTING MASS PROPERTIES

- **When the template is created, default mass properties are assigned to the bodies. You can modify these values in Standard Interface.**
- **To modify mass properties, from the Adjust menu, select General Part. You have two options:**
  - **Modify**
    - Displays a dialog box to select a part
    - Can specify mass and inertia values.
    - Can also display this dialog box by right-clicking on the part and selecting the part name followed by Modify.
  - **Calculate Mass**
    - Adams/Car calculates the new values for mass and inertia based on the Adams/Car geometry and the density.
    - Note that if the geometry is changed in Standard Interface from the template's default value, the respective part's mass will not automatically change.
    - To change it, simply use the Calculate Mass function again. If your geometry is imported from a CAD package and is complex, you may have to enter the mass manually.

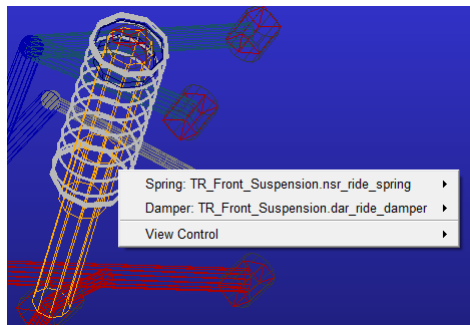
# ADJUSTING SPRINGS AND DAMPERS

- **A spring or a damper is created in Template Builder and references a property file located in a particular folder in your selected database.**
- **In Standard Interface, you can link the spring or damper to a different property file or you can create a new property file.**
- **To modify a spring, right-click the spring and select Modify.**
  - Or, From the Model Browser, point to **Force Elements> Springs/Dampers**, right-click the spring/damper you want to modify, select **Modify**. This displays the dialog box on the following page

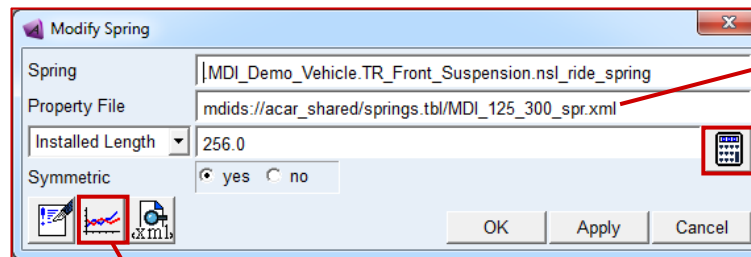
# ADJUSTING SPRINGS AND DAMPERS



Using Model Browser

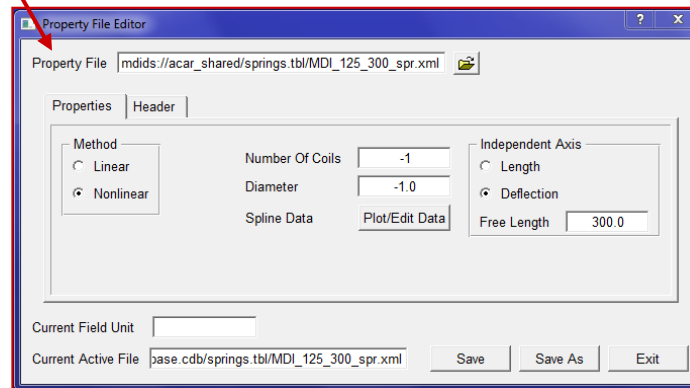


Using GUI



When right-click – Search, Adams/Car takes you to the spring.tbl directory in the selected database

Calculates the required installed length for a given preload



Tips: The units of the property file are automatically converted to the units of the model.

More information about property files on the next few pages

# PROPERTY FILES - INTRODUCTION

- **Property files are ASCII-based files that contain data for modeling components, such as springs, dampers and bushings. Because property files are flat text files, you can use any text editor to create or modify them.**
- **Property files allow you to:**
  - Easily apply the same characteristics or parameters to many components within a template or subsystem. In Adams/Car for example, a suspension might contain many bushings with the same properties. In this case, all the bushings could reference the same property file.
  - Share a component between different templates and subsystems.
- **All property file types are specified in the configuration file (acar.cfg). When you edit property files, you can save them either with the existing file name or with a new file name.**
- **Property files are grouped in classes and stored in databases. Every class (such as bushings and dampers) is filed in the corresponding database table.**

# PROPERTY FILES - INTRODUCTION

- There are two types of property files: TeimOrbit files and XML files. The TeimOrbit files are created/edited using the Curve Manager. The XML files are modified using the Property File Editor. TeimOrbit files have extensions based on property file types (e.g. .spr for spring, .bus for busing). Adams/Car provides tools to convert TeimOrbit files to XML files. (XML files are preferable in ADAMS 2017.2 due to the GUI difference)

MDI\_125\_300.spr

```
$-----  
[MDI_HEADER]  
FILE_TYPE      = 'spr'  
FILE_VERSION   = 4.0  
FILE_FORMAT    = 'ASCII'  
$-----  
[UNITS]  
LENGTH        = 'mm'  
ANGLE          = 'degrees'  
FORCE          = 'newton'  
MASS           = 'kg'  
TIME           = 'second'  
$-----  
[SPRING_DATA]  
FREE_LENGTH    = 300.0  
$-----  
[CURVE]  
{ disp      force}  
-100.0      -12500.0  
-90.0       -11250.0  
-80.0       -10000.0  
-70.0       -8750.0  
-60.0       -7500.0  
-50.0       -6250.0
```

MDI\_125\_300.xml

```
<?xml version="1.0" encoding="UTF-8"?>  
<afc xmlns="http://www.mscsoftware.com:/vfc" xmlns:afc="http://www.mscsoftware.com:/vfc">  
  <afc:Bibliography>  
    <File schema="afc" version="1.0.0.0" URI="/vobs/vpc/acar/tests/c/>  
    <Revision version="1">  
      <Comment>  
        <![CDATA[  
Equivalent to the shared car database spring file MDI_125_300  
]]>  
      </Comment>  
    </Revision>  
    <Corporation author="Mechanical Dynamics, Inc." URI="http://www.mscsoftware.com" />  
    <Author user="carlib" name="Car Library" />  
    <Environment hostName="yukon" operatingSystem="IRIX64 6.5" />  
    <Application name="ADAMS/Chassis" version="2003" />  
  </afc:Bibliography>  
  <afc:Units>  
    <afc:UnitSetting name="angle" current="deg" />  
    <afc:UnitSetting name="current" current="ampere" />  
    <afc:UnitSetting name="force" current="newton" />  
    <afc:UnitSetting name="length" current="mm" />  
    <afc:UnitSetting name="luminosity" current="candela" />  
    <afc:UnitSetting name="mass" current="kilogram" />  
    <afc:UnitSetting name="quantity" current="mole" />  
  </afc:Units>  
</afc>
```

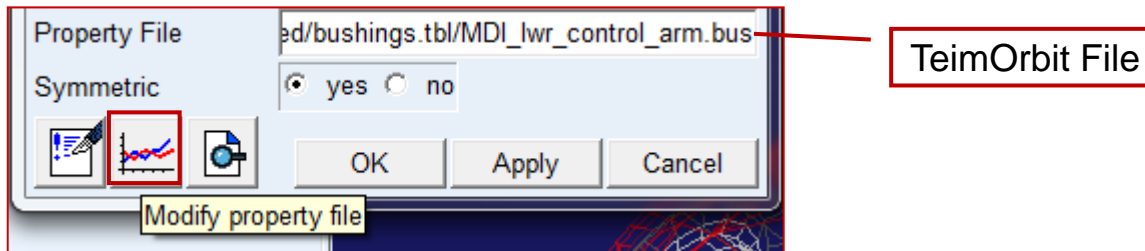


# PROPERTY FILES - CURVE MANAGER

- **The Curve Manager supports the following curve types:**
  - **Bushing** - Specify all six curves that define 3-D translational and rotational stiffness. Unlike a BUSHING statement, these curves can be nonlinear. A damping coefficient can also be included.
  - **Bumpstop** - A stiffness curve (force versus deflection)
  - **Reboundstop** - A stiffness curve (force versus deflection)
  - **Spring** - A stiffness curve (force versus deflection) and a free length (for information on springs, see Springs, in Section 11)
  - **Damper** - A damping curve (force versus velocity)
  - **Wheel envelope** - Input boundaries:
    - Steer input (length or angle)
    - Wheel interior points and boundary
    - Steer interior points and boundary
  - **Differential** - A viscous element characteristic curve (torque versus slip speed)
- **The functionality of the Curve Manager changes, depending on the kind of property file being used.**

# PROPERTY FILES - CURVE MANAGER

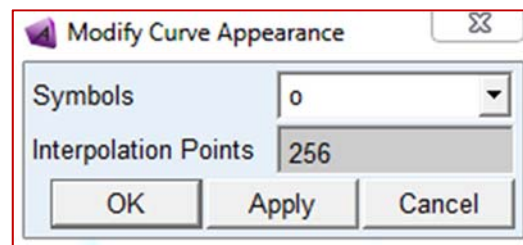
- To access Curve Manager, in either Standard Interface or Template Builder, from the **Tools** menu, select **Curve Manager**. You can also open it using the modify dialog box for parts referencing the TeimOrbit property files. (**Edit menu**→ **modify**→ **double click the specific part**)




- Features in the Curve Manager:**
  - Fit the curve on the plot
  - Zoom a part of the curve
  - Curve math
  - Vertical hotpoints
  - Toggle memory curve

# PROPERTY FILES - CURVE MANAGER

- **Curve Manager has two modes:**
  - **Plotting** - In this mode you can build a curve by specifying functions that define the curve. For example, you can define a spring curve with a rate of 20 N/mm with 25 points between -100 and 100 mm.
  - **Table** - In this mode you can specify each point in a data table. For example, for the same spring curve made in the plotting mode, you would have to type in the x-y numbers for all 25 points.
- **To switch between the plot and table format, use the View command on the main menu of the Curve Manager and select either Plot or Table.**
- **In the Plotting mode, you can change the symbols that represent the data points, from the Settings menu - Appearance.**



# PROPERTY FILES - CURVE MANAGER

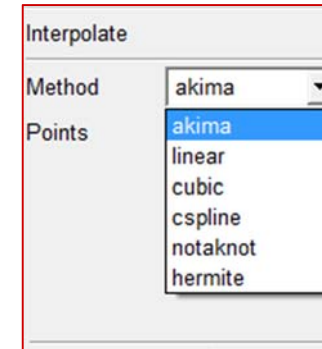
- **Curve Math**  (right click)
  - **Slope** - Specify a rate, limits, and number of points or number of segments, which is the same as number of points minus 1.
  - **Offset** - Offsets the curve by the value you specify.
  - **Absolute value** - No parameters, takes the absolute value of the curve.
  - **Negate** - Inverts the curve.
  - **Zero** - Offsets the curve so it starts from zero.
  - **Y mirror** - Mirrors about the y axis.
  - **XY mirror** - Mirrors the x and y values, so that the curve goes through the same value for both the x and y axis.
  - **Copy x → y** - Makes the y value the same as the x.
  - **User Defined Function** - Specify a function, limits and number of points or segments, and you'll get a curve of the function you specified.



# PROPERTY FILES - CURVE MANAGER

- **Interpolate** - Uses one of the following interpolation methods and creates the number of points you specify:

Akima	Cspline
Linear	Notaknot
Cubic	Hermite



- **Step** - Specify start and end values, and the y value for those start and end points.
- **Scale** - Scales the curve by the value you specify.
- **Ramp** - Specify start and end values and the y value for those start and end points.
- **Expand** - Stretches the x start and end points.
- **Sine** - Start and end points for x and y values, when the sweep starts, minimum and maximum amplitude, frequency and the number of points or segments.

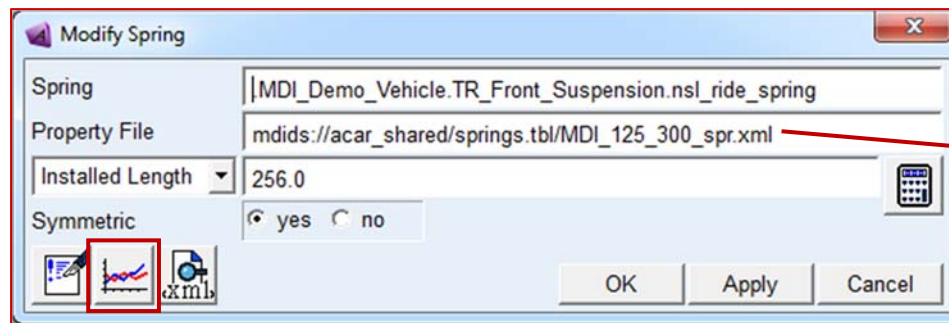
**Note: Curve math is not available for wheel envelope.**

# PROPERTY FILES - PROPERTY FILE EDITOR

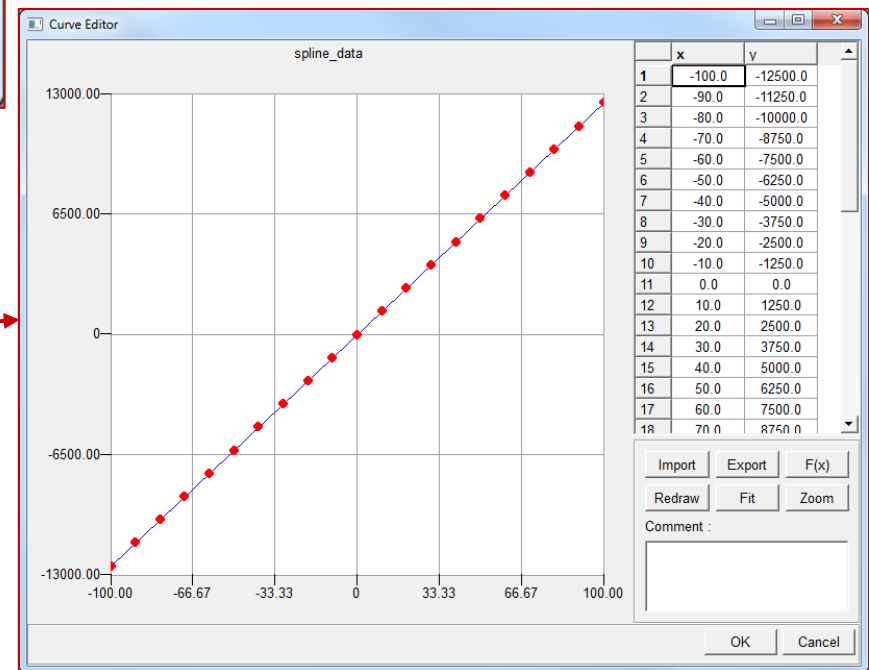
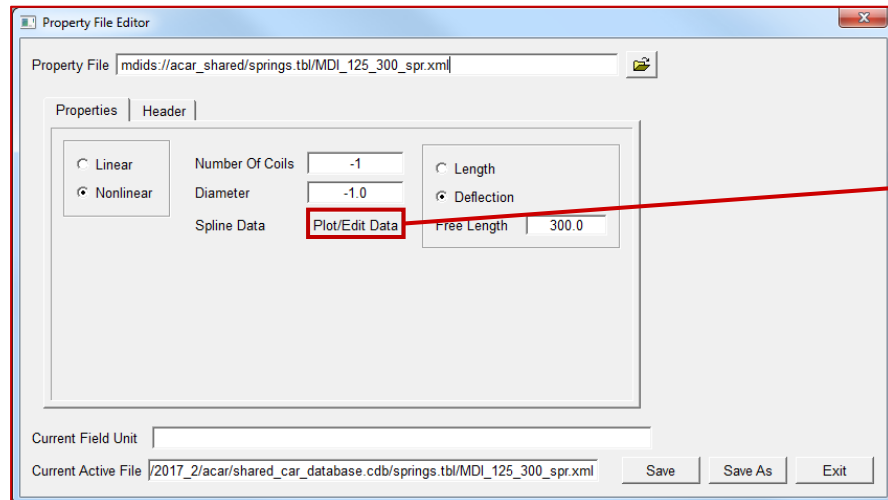
- **The Property File Editor allows you to edit component data and store that data in XML format. The following components can use this format:**
  - Air springs
  - Bumpstops
  - Bushings
  - Dampers
  - Reboundstops
  - Springs
- **See the acar\_shared database for example XML property files.**

# PROPERTY FILES - PROPERTY FILE EDITOR

- To access Property File Editor, in either Standard Interface or Template Builder, use the Modify option on parts referencing an XML property file.



XML File



# PROPERTY FILES - SAVING TO DEFAULT WRITABLE

- **When you finish editing the property file, you can save it. Adams/Car saves it to the corresponding table directory in the default writable database. For example, Adams/Car saves a spring property file to the table directory spring.tbl, in the database.**
- **To check what the default writable database is:**
  - From the Tools menu, point to Database Management and then select Database Info.
  - See the subtitle in the plot, which shows the complete path to the property file.



# PROPERTY FILES - CONVERTING TEIMORBIT TO XML

- You can use the conversion utility to convert either a single property file or an entire database from TeimOrbit to XML format (Tools → Database Management → Version Upgrade → TeimOrbit → XML).
- In the case of an entire database, the utility receives the database that you want to convert and the target database. All TeimOrbit property files are read, converted, and saved to the target database.
- In cases where data that is available in the XML file but was not defined (or was not supported) in the TeimOrbit file, a series of defaults have been used.

# REPLACING INSTANCE DEFINITION

- **Another way of modifying components is to change the type of definition used for that component. For example, you can replace a coil spring with an air spring.**
- **To change component definition:**
  - In Standard Interface, right-click a component, point to its name, and then select **Replace**.
  - Specify the definition you want to change into and if you want to replace only the selected instance or all the instances with the same name or type.
  - Select **OK**.
- **Components currently supported include: air spring, bushing, damper and spring.**
- **Note:** You can change the component definition only in the Standard Interface.

# MORE SUBSYSTEM ADJUSTMENTS

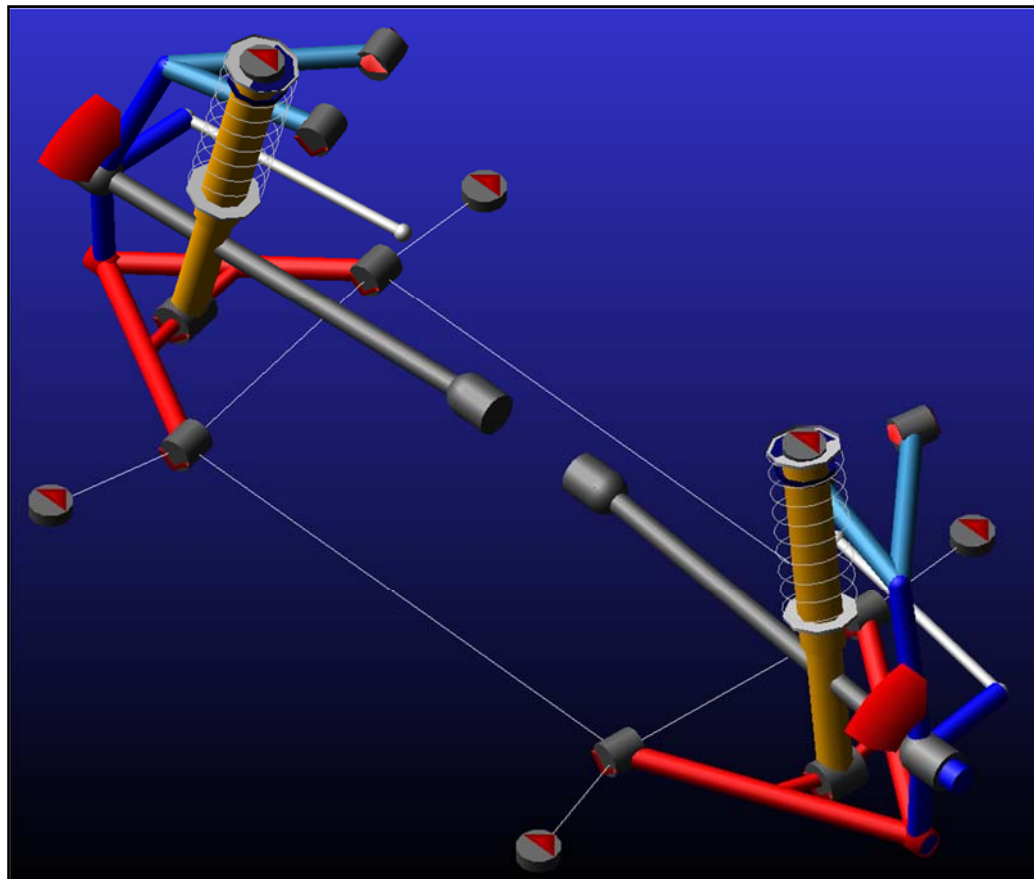
- **For more options, select View → Subsystem**
- **Suspensions**
  - Toggle driveline activity on/off
  - Suspension parameter array (in Toe/Camber Angles and Suspension Parameter Array, in Section 12)
  - Toe/camber values
- **Steering**
  - Gear ratios
  - Steering assist property file
- **Powertrain**
  - Powertrain map property file
  - Differential property file

# EXERCISE

- **Perform Workshop 3, “CREATING AND ADJUSTING SUBSYSTEMS”.**

# SECTION 4

## CREATING AND SIMULATING SUSPENSIONS



# CREATING AND SIMULATING SUSPENSIONS

- **In this section, you will learn how to create a suspension assembly in Adams/Car. You will learn about the available suspension analyses and how to submit them. You will also learn about the plot configuration files, how to create them and how to review the results of your analyses.**

# CREATING AND SIMULATING SUSPENSIONS

- **What's in this section:**
  - Creating Suspension Assemblies
  - Half-Vehicle Analyses
  - Suspension Parameters
  - Creating Loadcases
  - Dynamic Suspension Analysis
  - Warning Messages
  - Files Produced by Analyses
  - Plot Configuration Files

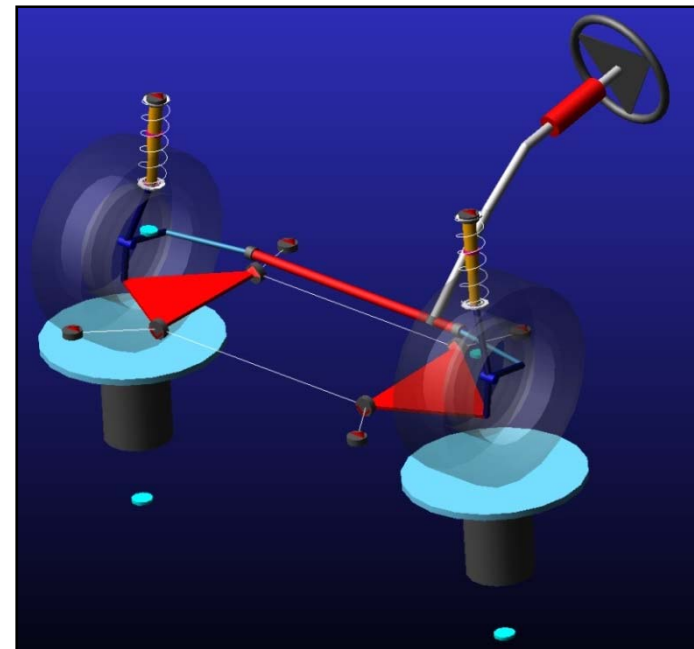
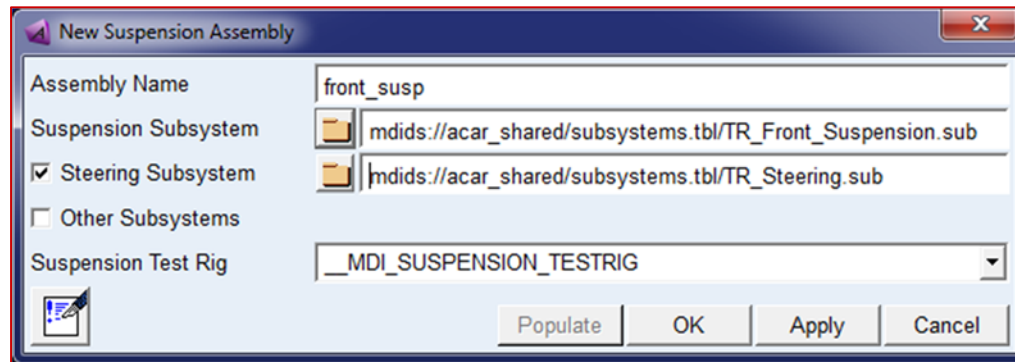
# CREATING SUSPENSION ASSEMBLIES

- **An assembly consists of a single test rig and one or more subsystems (a test rig by itself is just a specialized subsystem).**
- **You create suspension assemblies in Standard Interface: from the File menu, point to New, and then select Suspension Assembly. In the dialog box, specify all the subsystems to be included in the assembly, as well as the test rig.**
- **If you use subsystems created from new templates, you need to make sure the communicators match up. Communicators are special variables defined at the template level to bring each subsystem together, and with the test rig, to create an assembly (communicators are described in more detail in Section 13 – Communicators).**



# CREATING SUSPENSION ASSEMBLIES

- The picture below shows a suspension assembly containing a suspension and steering subsystem, and test rig.
- The test-rig controls the simulation; the suspension test-rig must appear in a suspension assembly.



# HALF-VEHICLE ANALYSES

- **You can perform the following types of suspension analyses in Adams/Car:**
  - **Parallel wheel travel** - Both wheels move in unison.
  - **Opposite wheel travel** - Wheels move out of phase.
  - **Roll and vertical force** - Sweeps roll angle while allowing wheels to find their own vertical position, the total force on both wheels is held constant.
  - **Single wheel travel** - One wheel fixed, while other moves.
  - **Steering** - Motion applied to steering wheel or rack.

# HALF-VEHICLE ANALYSES

- **Static load** - Force or displacement applied at specified locations (wheel center, tire contact patch), with or without steering
- **Dynamic Analysis** - Apply test data to drive the test rig actuators and carry out dynamic analysis
- **External files:**
  - **Loadcase** – specify a file that describes a custom loading scenario. Loadcase files can be created using the Generate Loadcase dialog box
  - **Wheel envelope** – a series of parallel wheel actuations in which the steer angle is being swept over a specified range. Generates a table of points that define the wheels position for wheel interference/collision analysis

(When performing a suspension analysis, Adams/Car uses the first second to bring the wheel centers to the lowest position, and then uses as many seconds as you specify steps, to move the suspension to the upper position.)



**For more information, see the Running Analyses section in the Adams/Car online help.**

# SUSPENSION PARAMETERS

- Some of the default request outputs need information that is not available in the model. Therefore, **you must supply this additional information**. This information has no bearing on the outcome of the simulation, and **only affects the values of some “user-defined” (pre-defined) results (for example, the roll center)**.
- Adams/Car stores the input in an array named Suspension Parameters, which you can find in the Standard Interface under Simulate → Suspension Analysis → Set Suspension Parameters.
- **The values you must supply are:**
  - Loaded tire radius
  - Tire stiffness (user-entered or from property file)
  - Sprung mass

# SUSPENSION PARAMETERS

- **CG height**
- **Wheelbase**
- **Drive ratio**
- **Brake ratio**
- **Adams/Car uses the parameters `cg_height`, `wheelbase`, `sprung_mass`, and `tire stiffness` when calculating:**
  - Percent anti-dive
  - Dive
  - Percent anti-lift
  - Lift
  - Percent anti-squat
  - Roll center height
  - Suspension rates
  - Total roll rate

**Suspension Analysis: Setup Parameters**

Suspension Assembly: `mdi_front_vehicle`

**Suspension Settings**

Tire Model: `User Defined`

Tire Unloaded Radius: `300.0`

Tire Stiffness: `200.0`

Tire Property File: `RIGID_WHEEL`

Wheel Mass: `1.0`

Dual Wheels: ☐ Active ☒ Inactive

Dual Wheel Offset: `300.0`

**Vehicle Parameters**

Sprung Mass: `1200.0`

CG Height: `300.0`

Wheelbase: `2000.0`

Drive Ratio (% Front): `60`

Brake Ratio (% Front): `55`

Buttons: OK Apply Cancel

# CREATING LOADCASES

- **A loadcase is an ASCII file containing all necessary information to run a simulation. It is basically a way of scripting suspension simulations with these five force/actuation types:**
  - Wheel travel (Parallel, Opposite, Single)
  - Steering
  - Static load
  - Roll and vertical force
- **You can call many loadcases, and Adams/Car will run them one at a time.**
- **You can create a loadcase file by selecting Simulate → Suspension Analysis → Create Loadcase. Then, select the type of analysis you want to run, and specify the relevant data.**

# DYNAMIC SUSPENSION ANALYSIS

- A ‘Dynamic’ option is also available under Suspension Analyses.
- Many suspension analyses are of type “quasi-static”; dynamic analyses capture the effect of transient forces and phenomena such as damping.
- The Suspension assembly can be excited by providing input as displacement, velocity, acceleration or Force to the Jacks (defined in the testrig).
- Use a RPC File to drive the testrig using test data or use runtime View functions.

# WARNING MESSAGES

- When you create an assembly, you will sometimes see warning messages.
- The models in the shared car database contain all the communicators that could possibly be used by other systems, and in many cases, not all communicators are used. The Message window displays the following:

```
Creating the suspension assembly: 'assembly'...
Opening the rear suspension subsystem: 'my_macph'...
Assembling subsystems...
Assigning communicators...
WARNING: The following input communicators were not assigned during assembly:
    testrig.cil_jack_frame (attached to ground)
    testrig.cir_jack_frame (attached to ground)
    testrig.cis_leaf_adjustment_steps
    testrig.cis_powertrain_to_body (attached to ground)
    testrig.cis_steering_rack_joint
    testrig.cis_steering_wheel_joint
    my_macph.cil_tierod_to_steering (attached to ground)
    my_macph.cir_tierod_to_steering (attached to ground)
    my_macph.cis_subframe_to_body (attached to ground)
    my_macph.cil_strut_to_body (attached to ground)
    my_macph.cir_strut_to_body (attached to ground)
    my_macph.cil_ARB_pickup
    my_macph.cir_ARB_pickup
Assignment of communicators completed.
Assembly of subsystems completed.
Suspension assembly ready.
```

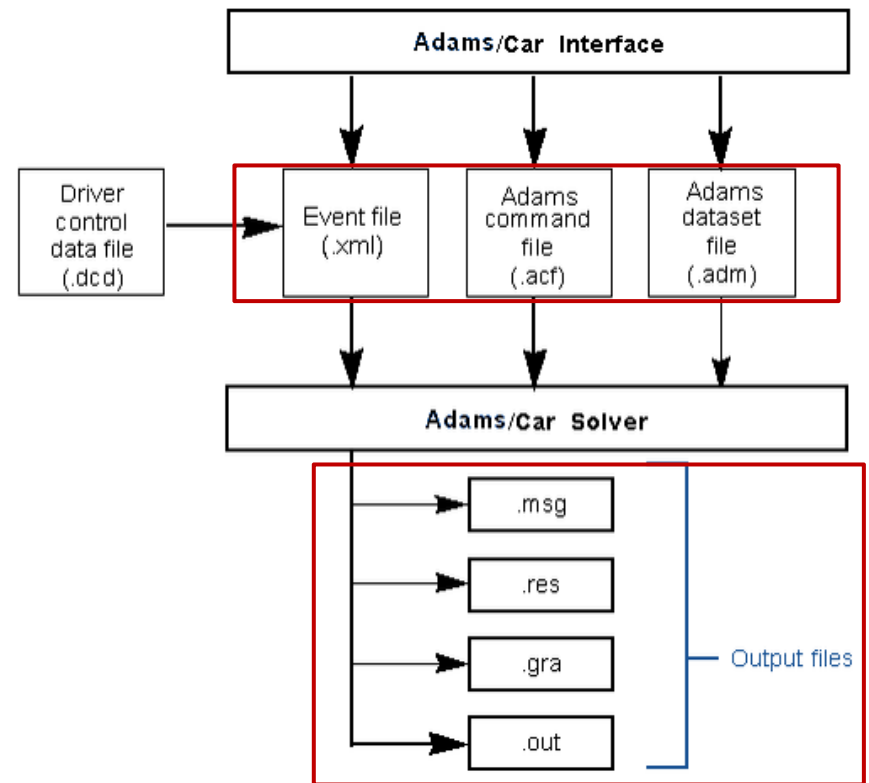


# WARNING MESSAGES

- **If Adams/Car cannot find a matching communicator, it attaches it to ground.**
- **Look at communicator warning messages to make sure they make sense. Ensure that communicators of interest are not attached to ground.**

# FILES PRODUCED BY ANALYSES

- Many files are created in the working directory, such as:
  - Input files: (.adm) model file, (.acf) command file, event and other input files.
  - Output files: message (.msg), request (.req), results (.res), graphics (.gra), and output (.out) files.



# PLOT CONFIGURATION FILES

- **Plot configuration files store your custom plot layouts. You can apply a plot configuration file to subsequent analyses to quickly restore things like:**
  - titles, labels, horizontal and vertical spacing's, scaling, legend text, and more
- **You can cross-plot multiple analyses of the same type using one plot configuration file.**
- **Plot configuration files are stored in your database in the plot\_configs.tbl directory.**
- **You can access plot configuration files from Adams/PostProcessor.**

# PLOT CONFIGURATION FILES

- **To create a plot configuration file:**
  - Create and configure plots as desired, including specifying labels and spacing.
  - From the **File** menu, point to **Export**, and then select **Plot Configuration File**.
  - In the **Configuration File Name** text box, enter the name for the **plot configuration**.
  - Select the plots and curves that you want to include then select **OK**.
- **To use a plot configuration file:**
  - **Create/Import** a new analysis
  - From the **Plot** menu, select **Create Plots**.
  - In the **Analyses** text box, enter the analysis or analyses from which you want to view results.
  - In the **Plot Configuration File** text box, select the plot configuration file and hit **OK**.

# PLOT CONFIGURATION FILES

- After your template-based product creates each curve, it executes the following commands if you defined a command keyword:

```
acar custom_plots <command_keyword> &  
analysis=<analysis> &  
plot_name=<plot_name> &  
vertical_data=<y> &  
horizontal_data=<x> &  
curve_name=<curve_name>
```

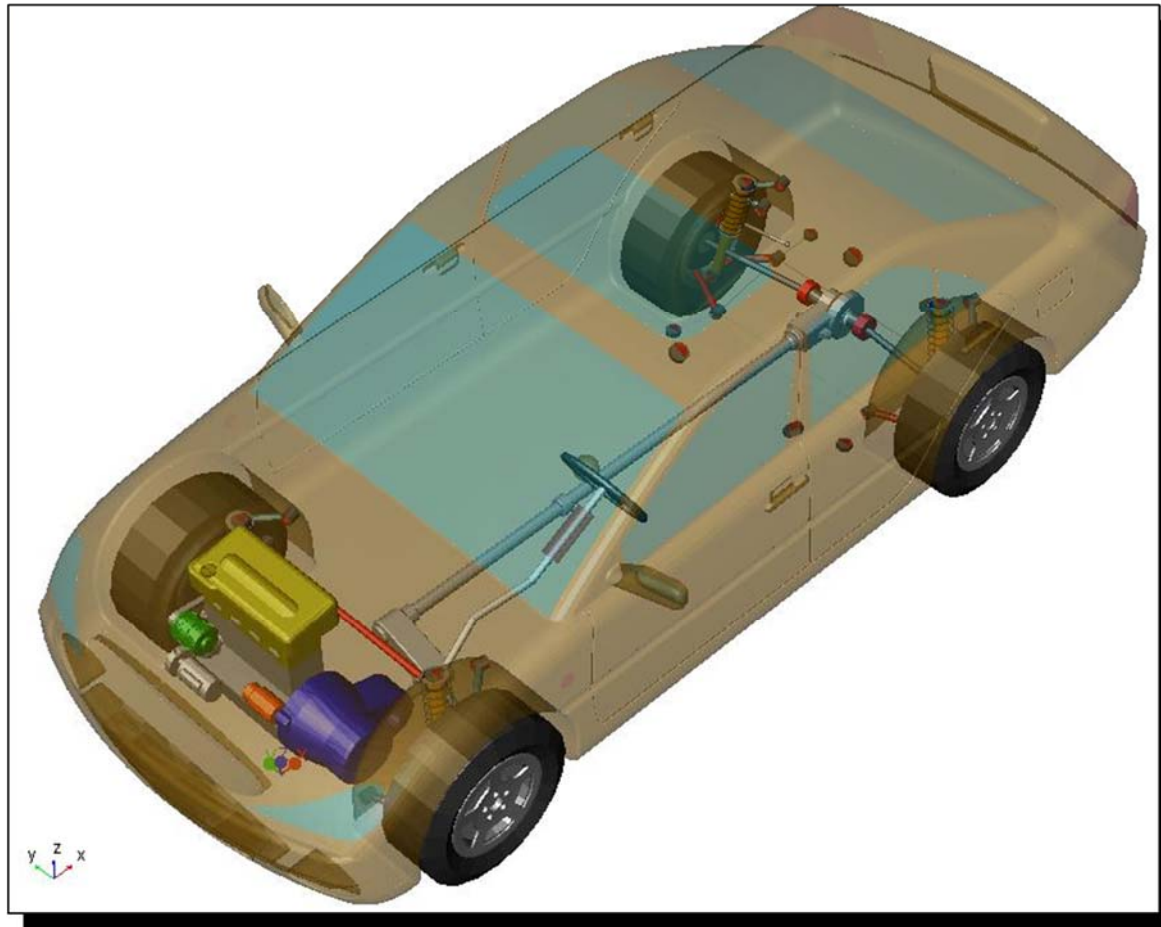
- The command `acar custom_plots <command_keyword>` must already be created in the current session, either interactively or already present in the `acar.bin` file.
- See KB8014901 (Adding titles and note to plots with the plot configuration file) at:  
<http://simcompanion.mscsoftware.com/infocenter/index?page=content&id=KB8014901>
- We provide several example plot configuration files in the Adams/Car shared database. For details of the output they plot, see KB8016050 at:  
<http://simcompanion.mscsoftware.com/infocenter/index?page=content&id=KB8016050>

# EXERCISE

- **Perform Workshop 4, “RUNNING SUSPENSION ANALYSIS”.**

# SECTION 5

## IMPORTING CAD GEOMETRY



# IMPORTING CAD GEOMETRY

- **In this section, you will learn how to import geometry (graphics) files in Adams/Car.**



# IMPORTING CAD GEOMETRY

- **CAD geometry can be brought into Adams/Car**
- **Users can import file formats supported by Adams such as Parasolid, IGES, stl, slp, obj, etc. Each file format may have slightly different information required or options provided -- items like scaling, reference marker, rotation, translation, etc.**
- **Common formats such as shell (.shl) and stereolithography (.stl) provide effective models, and they can be generated from most CAD packages. (.stl) format will have sharper images and animations than (.shl) format.**

# IMPORTING CAD GEOMETRY

- **To use a shell file (.shl) for adding graphics (geometry), the recommended approach is to first create a part through the Build Menu. Then add the geometry to this part (File → Import).**

(This is a little different from Adams/View in which you let Adams/View import the geometry and create the part in a single step. The reason is that creating parts via the Template Builder's Build Menu creates the necessary parameterization)

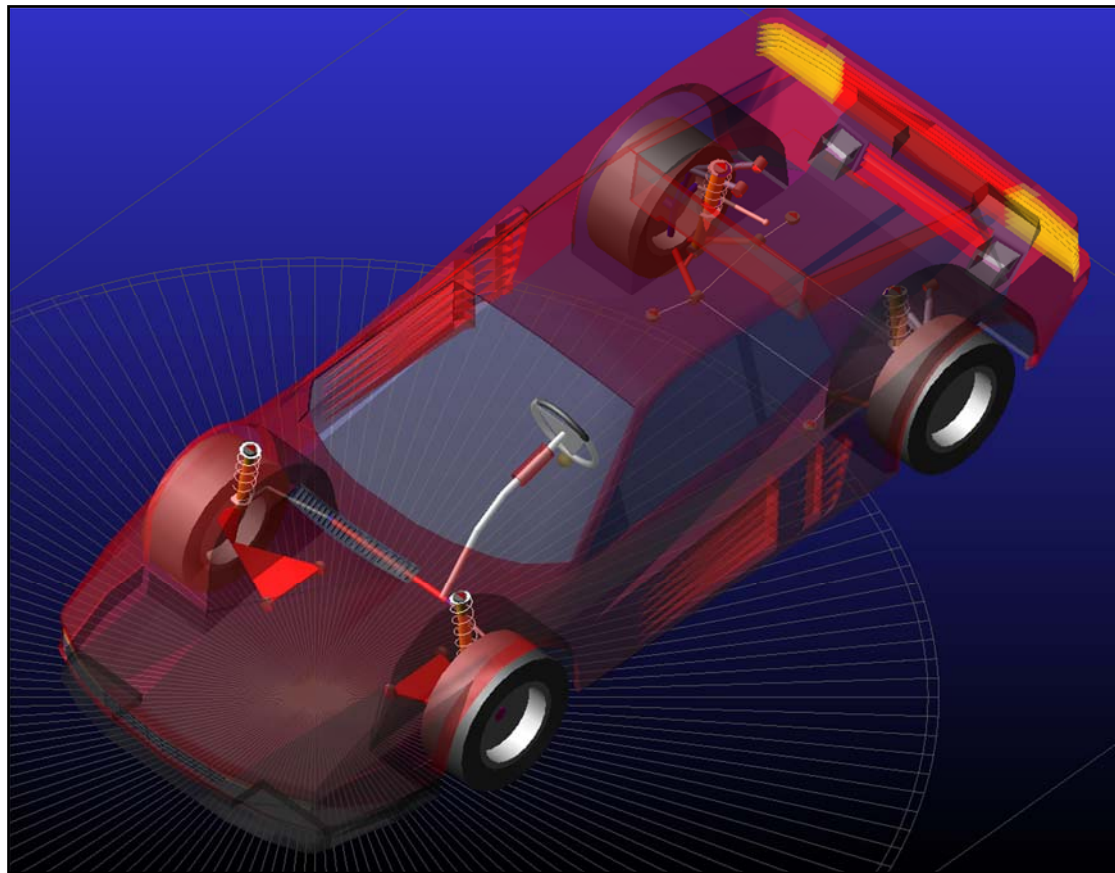
# EXERCISE

- **Perform Workshop 5, “IMPORTING CAD GEOMETRY”.**



## SECTION 6

# CREATING AND SIMULATING FULL VEHICLES



# CREATING AND SIMULATING FULL VEHICLES

- **In this section, you will learn how to create full-vehicle assemblies in Adams/Car. You will also learn about full-vehicle analyses and static Vehicle Setup analysis.**

# CREATING AND SIMULATING FULL VEHICLES

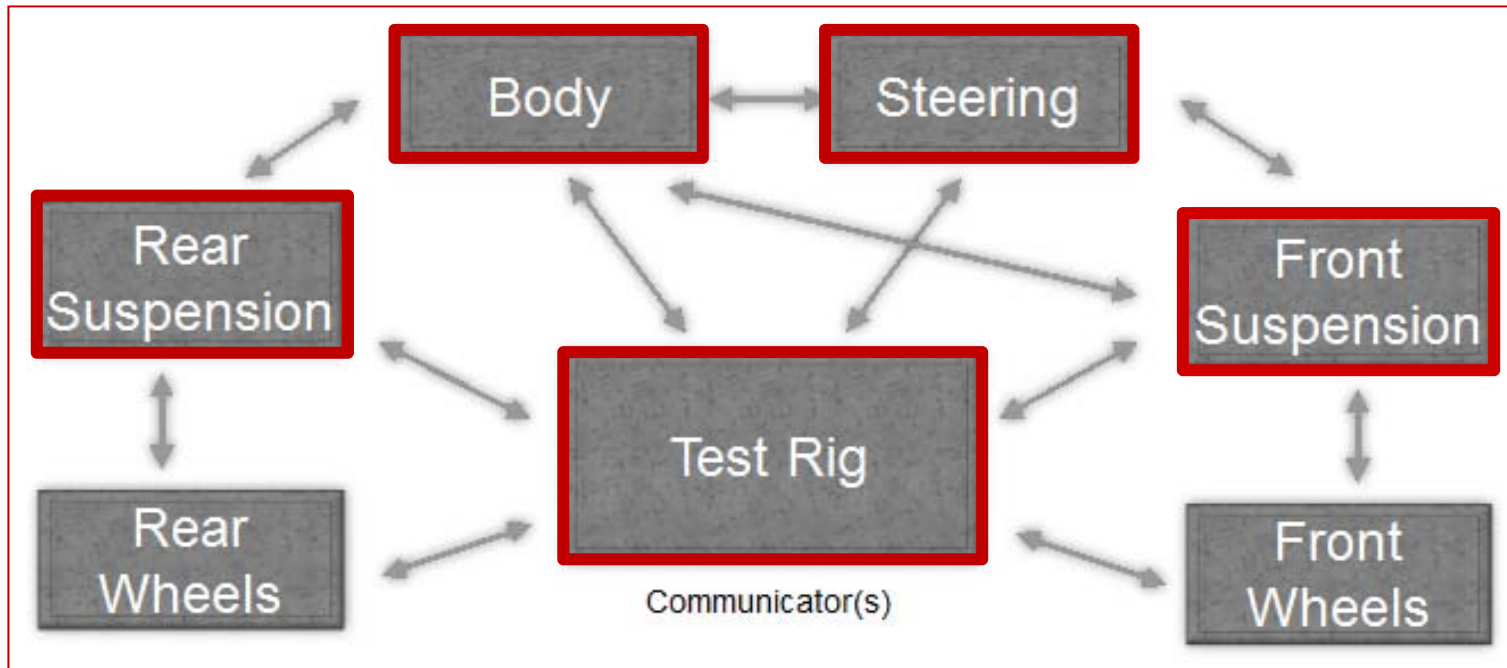
- **What's in this section:**
  - Creating Full-Vehicle Assemblies
  - Shifting Subsystems
  - Updating Subsystems and Assemblies
  - Synchronizing Subsystems and Assemblies
  - Adding/Removing and Activating/Deactivating Subsystems
  - Adjusting Mass
  - Static vehicle set-up

# CREATING FULL-VEHICLE ASSEMBLIES

- **A full vehicle assembly includes at least the following subsystems:**
  - Body
  - front and rear suspension
  - front and rear tires
  - steering system
  - a test rig to provide inputs to the virtual vehicle
- **To create a full-vehicle assembly: File → New → Full Vehicle.**
- The provided `.__MDI_SDI_TESTRIG` contains the Driving Machine allowing for open-loop, closed-loop, and quasi-static analyses
- Additional subsystems can be included via the Other Subsystem field

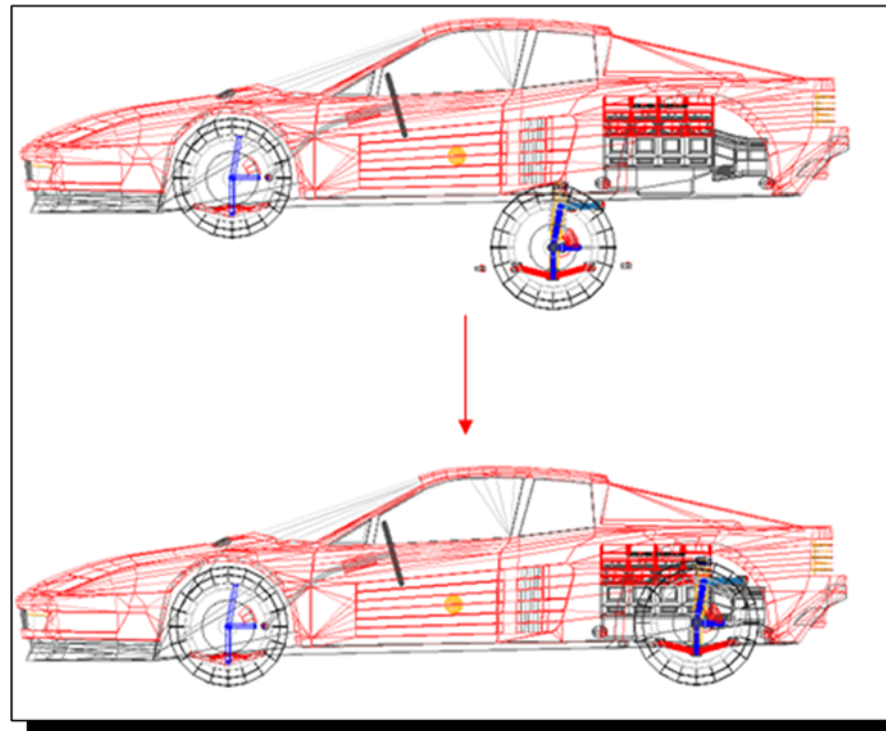


# FULL VEHICLE ASSEMBLY ILLUSTRATED



# SHIFTING SUBSYSTEMS

- To shift a subsystem, in the Standard Interface go to Adjust → Shift
- Here you have the choice of shifting the subsystem fore/aft and up/down. Fore/aft moves the subsystem along the global x-axis. Up/down moves it along the global z-axis



# UPDATING SUBSYSTEMS AND ASSEMBLIES

- **Updating subsystems**

- You may want to change a subsystem that is used within your assembly. To do this, go to File → Manage → Subsystem → Update, and select the subsystem.

Note: You must save the changed subsystem in the database to be loaded into the assembly. Also, the updated subsystem changes do not affect topology; the parameters that can be adjusted at the subsystem level are updated.

- **Updating assemblies**

- Similar to updating subsystems, you can update your assembly. To do this, go to File → Manage → Assembly → Update, and then select the assembly.

This option prevents you from having to close the assembly and reopen it with the modified subsystems by reloading information from the saved subsystems that the assembly references.

# SYNCHRONIZING SUBSYSTEMS AND ASSEMBLIES

- **Adams/Car allows you to automatically synchronize multiple in-session versions of the same subsystem. You can select a master subsystem (source) and choose from a dynamically updated list of other subsystems (targets) in the session.**
- **The target subsystems then inherit all settings of the master (only the subsystem mode is retained), which helps alleviate some subsystem consistency concerns raised when we introduced support for multiple assemblies in the same session.**
- **For example, if a suspension assembly and a full-vehicle assembly both reference the same steering subsystem, a change to the steering wheel position in the suspension assembly will allow you to synchronize the change into the full-vehicle assembly.**

# ADDING/REMOVING AND ACTIVATING/DEACTIVATING SUBSYSTEMS

- **In assembly, you can modify the underlying subsystems even after the assembly is opened. You can perform five basic tasks on an assembly open in an Adams/Car session, as explained next.**
- 1. Add a new subsystem into an existing assembly:**
    - **File → Manage → Assemblies → Add Subsystem**
    - The assembly is first disassembled, which includes 'unassigning' communicators. Next, the new subsystem is opened underneath the existing assembly. The assembly is then re-assembled including re-assigning the communicators.

# ADDING/REMOVING AND ACTIVATING/DEACTIVATING SUBSYSTEMS

## 2. Replace an existing subsystem in an assembly with a new subsystem:

- **File → Manage → Assemblies → Replace Subsystem**
- The assembly is first disassembled, which includes 'unassigning' communicators. The existing subsystem is then deleted from the assembly. Next, the new subsystem is opened underneath the existing assembly. The assembly is finally re-assembled, which includes re-assigning the communicators.

## 3. Remove an existing subsystem from an assembly:

- **File → Manage → Assemblies → Remove Subsystem**
- The assembly is first disassembled, which includes 'unassigning' communicators. The existing subsystem is then deleted from the assembly. The assembly is finally re-assembled, which includes re-assigning the communicators.

# ADDING/REMOVING AND ACTIVATING/DEACTIVATING SUBSYSTEMS

## 4. Deactivate an existing subsystem in an assembly:

- **File → Manage → Assemblies → Toggle Subsystem Activity**
- The assembly is first disassembled, which includes 'unassigning' communicators. The existing subsystem is then deactivated, which means that it is not actually removed from the assembly, but simply ignored.

## 5. Activate an existing subsystem in an assembly:

- **File → Manage → Assemblies → Toggle Subsystem Activity**
- The assembly is first disassembled, which includes 'unassigning' communicators. The existing subsystem is then activated, which means that it will now be considered a valid part of the assembly.

# ADJUSTING MASS

- **Adjusting the mass of your vehicle automatically**
  - In Adams/Car you can adjust the mass properties of an assembled model. To adjust the aggregate mass, enter the desired mass, the desired centroidal inertias, and the desired center-of-mass location, relative to a marker. You also select a part that Adams/Car modifies to match the desired mass properties. Therefore, the mass properties of the virtual vehicle match those of the real vehicle.
  - To adjust the mass properties, go to **Simulation → Full Vehicle Analysis → Vehicle Set-up → Adjust Mass**.
  - To get the mass of some or all of the parts in your model, select **Tools → Aggregate Mass** (use the dialog box help (F1) for help on Aggregate Mass).
  - If the desired change causes the part you select to have a negative  $I_{xx}$ ,  $I_{yy}$ , or  $I_{zz}$  then the system will be unstable. To avoid, you should first reduce inertia values in other parts so that all parts satisfy the  **$I_{xx} + I_{yy} > I_{zz}$  criteria** without having negative inertias.



# ADJUSTING MASS

- **Adjusting the mass of your vehicle based on a typical measurement**
  - In most cases, the vehicle's center of gravity (CG) location data and inertia tensor are given with respect to the vehicle's CG. However, when you use **Tools → Aggregate Mass** with respect to the ground (default) reference frame, the inertia tensor will be reported with respect to the global coordinate system at (0,0,0), since you don't know where the vehicle CG is beforehand.
  - So, you have to first find out the vehicle CG location and create a CG marker in the template at that location. Then using **Tools → Aggregate Mass** again, find out the inertia tensor with respect to the CG . This will allow you to compare your Adams model mass properties to your measured data.

# STATIC VEHICLE SET-UP

- **The Static Vehicle Set-Up analysis sets some realistic adjustments for your vehicle, such as:**

1. Suspension Alignment
2. Weight Adjustment
3. Ride Height Adjustment

## **1. Suspension Alignment:**

- For the automatic suspension alignment during initial static equilibrium, adjustable forces are used.
- Parameter variables used to control the activity of the toe and camber adjusters are activated before the alignment (inactive by default) and automatically turned off at the end, which effectively replaces the adjustable forces with motion locks that maintain determined alignment settings.

# STATIC VEHICLE SET-UP

## 2. Weight Adjustment:

- Weight adjustment is done to achieve the desired corner weights.
- To accomplish this, the overall weight is adjusted by modifying pre-defined trim mass. Also its CG is moved to achieve the correct front/rear and left/right total weights.
- This adjustment is performed in a loop until the average change in corner weight error is less than the tolerance times the weight scale, or the specified number of iterations has been reached.
- After the first loop is complete, spring preload is then used to achieve a diagonal weight bias. The user has the option of selecting which spring preload is to be adjusted.

# STATIC VEHICLE SET-UP

## 3. Ride Height Adjustment:

- To adjust the ride height to a desired value user has to pre-define the four corner markers that will be used to measure ride height relative to the ground.
- The desired heights of these markers are entered and adjustment is performed in a loop by modifying spring preloads until the sum of the ride height errors is less than the corner height tolerance.

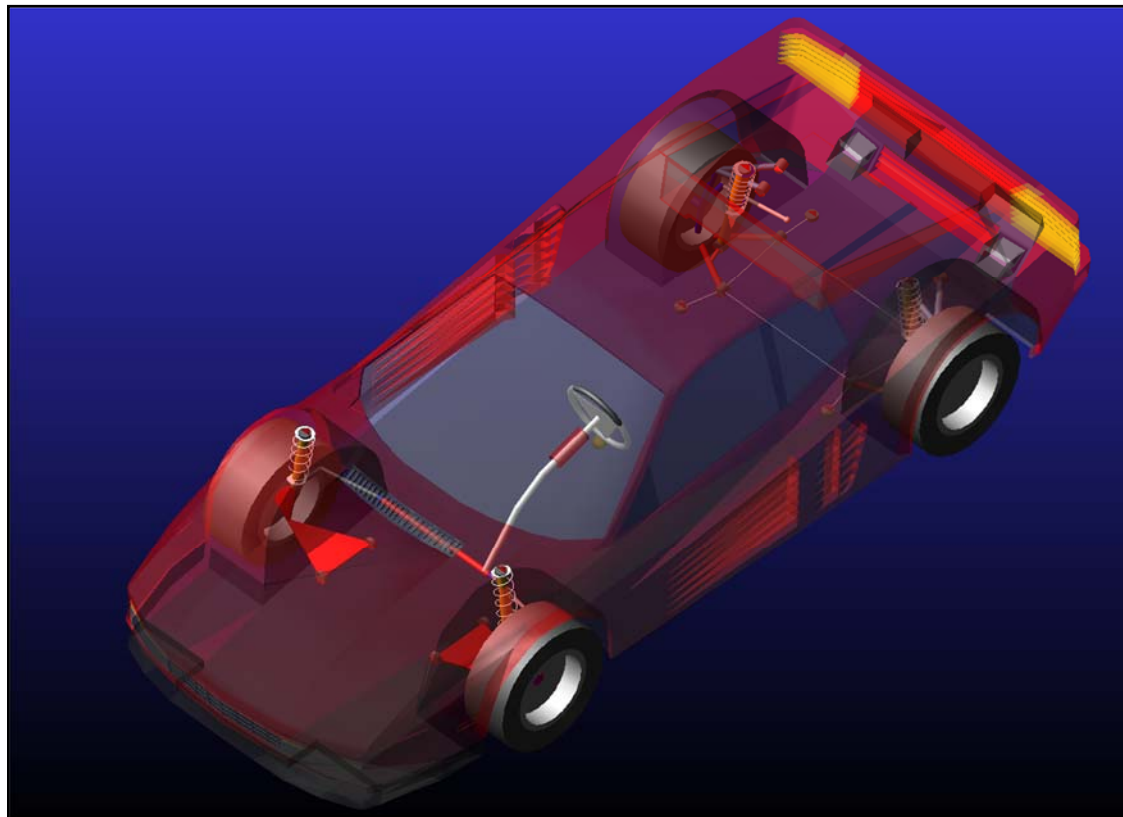
# EXERCISE

- **Perform Workshop 6, “RUNNING FULL-VEHICLE ANALYSIS AND ADJUSTING MASS”.**



# SECTION 7

## DRIVING MACHINE



# DRIVING MACHINE

- **The Driving Machine drives your virtual vehicle according to your instructions much like a test driver would drive an actual vehicle.**



# DRIVING MACHINE

- **What's in this module:**
  - Standard Driver Interface (SDI) and Driving Machine
  - Using the Driving Machine
  - Data Flow
  - Event File Details
  - Example Event .xml File
  - Driver Control Data Files
  - Creating .dcd Files

# STANDARD DRIVER INTERFACE (SDI) AND DRIVING MACHINE

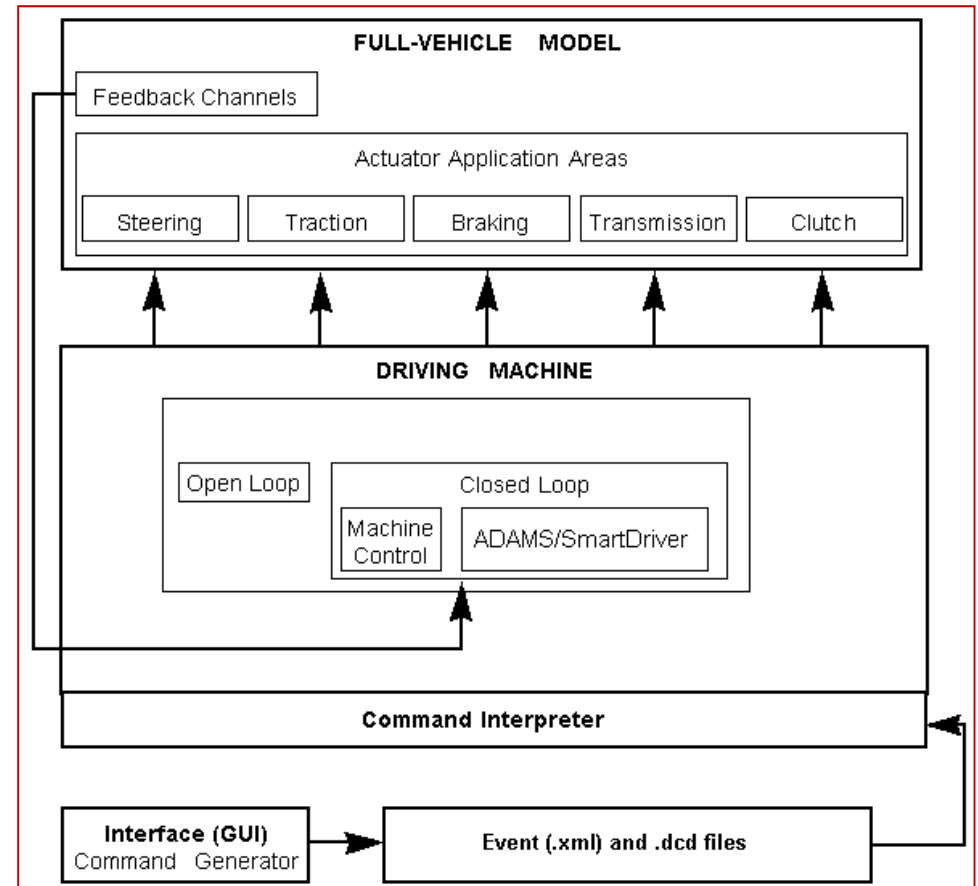
- **The Standard Driver Interface (SDI) provides the steering, throttle, clutch, gear, and brake inputs to your full vehicle assembly.**
- **You provide the inputs to the Driving Machine via either:**
  - **Open-loop control** - The open-loop controller can use either constant values or function expressions to drive the vehicle. No feedback is passed back to the controller.
  - **Machine control** - The machine controller is a closed-loop controller that controls the vehicle by using the vehicle states.
- **You can easily recreate physical test procedures or replay actual driving events using measured data.**

# USING THE DRIVING MACHINE

- **Using the Driving Machine, you can:**
  - Input the vehicle path  $\{x, y\}$  and speed. Use closed-loop, machine control **to steer a vehicle along a path** or **follow a desired velocity** or both.
  - Input a variety of **open-loop functions**, such as swept-sine and impulse, to the steering, throttle, or brake.
  - Input recorded steering, throttle, brake, gear, and clutch signals to your model.
  - Stop a simulation, switch controllers, and change output step size based on reaching a target lateral acceleration, longitudinal velocity, or distance traveled.

# USING THE DRIVING MACHINE

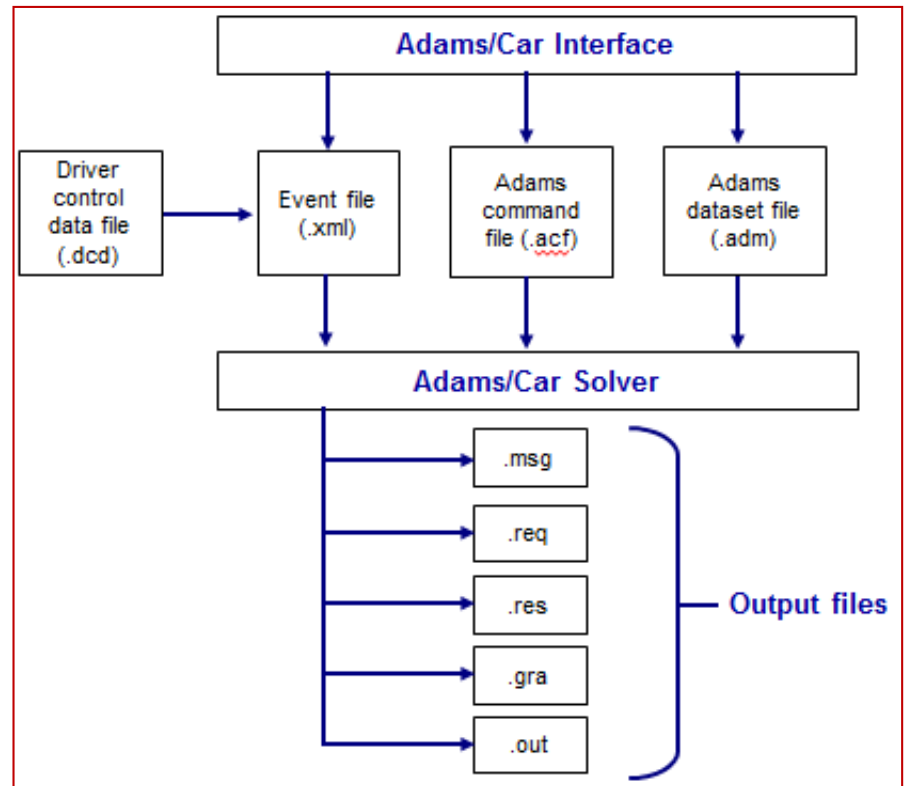
- **Steps in using the Driving Machine:**
  - Using the Driving Machine involves the following steps:
    - Assemble a full-vehicle model with the `.__MDI_SDI_TESTRIG` test rig or open an existing full-vehicle assembly.
  - Do either of the following:
    - Use a predefined analysis.
    - Create your own `.xml` files to perform your specific set of simulations.



# DATA FLOW

- **About data flow in the Driving Machine.**
  - Drive control inputs are stored in the event file (xml) and driver control (dcd) files.
  - The event file describes the maneuver(s) you want to perform
  - The event file can reference dcd files

*Note: Event and DCD file structure has been detailed in Appendix C*



# EVENT FILE DETAILS

- **The event file (.xml) instructs the Driving Machine how fast to drive the vehicle, where to drive the vehicle and when to stop the maneuver.**
- **Event files organize complex driving scenarios into a set of smaller, simpler steps called mini-maneuvers.**
- **Each maneuver can use different forms of control for the steering, throttle, brake, gear, and clutch signals.**
- **The event file also defines the static-setup method used to initialize the vehicle.**
- **The easiest way to create an event is to run a standard Full Vehicle Analysis and then edit the xml file in the Event Builder.**

# EXAMPLE EVENT FILES

The left screenshot shows the 'Event Builder' window with the 'Static Set-up' tab selected. It displays a table of mini-manuevers. A red box highlights the table, and a red arrow points from the 'turn\_in' row to the right screenshot.

Name	Active	Abort Time	Step Size	Sample Period
brake	yes	20.0	0.01	0.01
turn_in	yes	20.0	0.01	0.01
power_on_turn	yes	20.0	0.01	0.01
turn_out	yes	20.0	0.01	0.01
power_on_turn_2	yes	20.0	0.01	0.01
turn_out_2	yes	20.0	0.01	0.01

The right screenshot shows the 'Event Builder' window with the 'Static Set-up' tab selected. It displays the detailed configuration for the 'turn\_in' mini-manuever. A red arrow points from the 'turn\_in' row in the left screenshot to this window.

Static Set-up | Gear Shifting | Controller | Trajectory Planning | PID Speed & Path | PID Steering Output | Filters

Task: settle Initial Throttle: 0.0 Initial Brake: 0.0  
Halt On Failure: yes Initial Clutch: 0.0 Initial Steer: 0.0  
Linear: no Damping: yes

Name Filter: \*

Name: turn\_in Comment:

Steering | Throttle | Braking | Gear | Clutch | Conditions | Linear

Actuator Type: rotation Control Method: open Control Type: step  
☒ Absolute ☐ Relative

Start Time: 0.0 Duration: 0.5 Final Value: 6.981

Current Field Unit: angle (degree) Save and Run Save Save As Cancel

Mini-manuevers

Details of a mini-manuever

# DRIVER CONTROL DATA FILES

- Driver control data files contain data for use by the Driving Machine.
- To instruct the Driving Machine to use the data from a .dcd file, you must reference the file in a event file (.xml).
- You can reference the .dcd file for either open loop control or machine control as shown below:

This screenshot shows the configuration interface for a driver control data file. On the left, there are three dropdown menus: 'Actuator Type' set to 'rotation', 'Control Method' set to 'open', and 'Control Type' set to 'data\_driven'. Below these is a 'Control Mode' section with two radio buttons: 'Absolute' (selected) and 'Relative'. On the right, there is a 'Dcd Filename' text field containing the path 'shared\_car\_database.cdb/driver\_data.tbl/data\_driven.dcd' and a file selection icon.

This screenshot shows the configuration interface for a driver control data file. On the left, there are three dropdown menus: 'Actuator Type' set to 'rotation', 'Control Method' set to 'machine', and 'Control Type' set to 'data\_driven'. Below these is a 'Control Mode' section with two radio buttons: 'Absolute' (selected) and 'Relative'. On the right, there is a 'Steer Control' dropdown menu set to 'file', and a 'File Name' text field containing the path 'se.cdb/driver\_data.tbl/iso\_lane\_change.dcd' with a file selection icon.



# DRIVER CONTROL DATA FILES

- **Driver control data files hold two types of data, as follows:**
  - Open-loop data
    - Data that is played back as input to the vehicle without concern for how fast or where the vehicle goes. Such data includes: steering wheel angle, throttle, brake, gear, and clutch signals.
    - Examples of open-loop data include steering wheel angle versus time, and throttle position versus time.
  - Closed-loop data
    - Data that specifies exactly where and how fast the vehicle should go. An example of closed-loop data is vehicle x and y position versus time.
    - You may specify closed-loop data in several forms. For example, curvature and velocity versus distance travelled, or lateral acceleration and longitudinal acceleration versus time.
    - You specify the type of data using the `SPEED_CONTROL` and `STEERING_CONTROL` attributes in the `[CLOSED_LOOP]` section in the **.dcd file**.

# CREATING .DCD FILES

- **There is no tool available for creating or editing dcd files. However since they are ASCII files you can create and modify them using any text editor.**
- **Examples are available in the acar\_shared database under driver\_data.tbl table.**

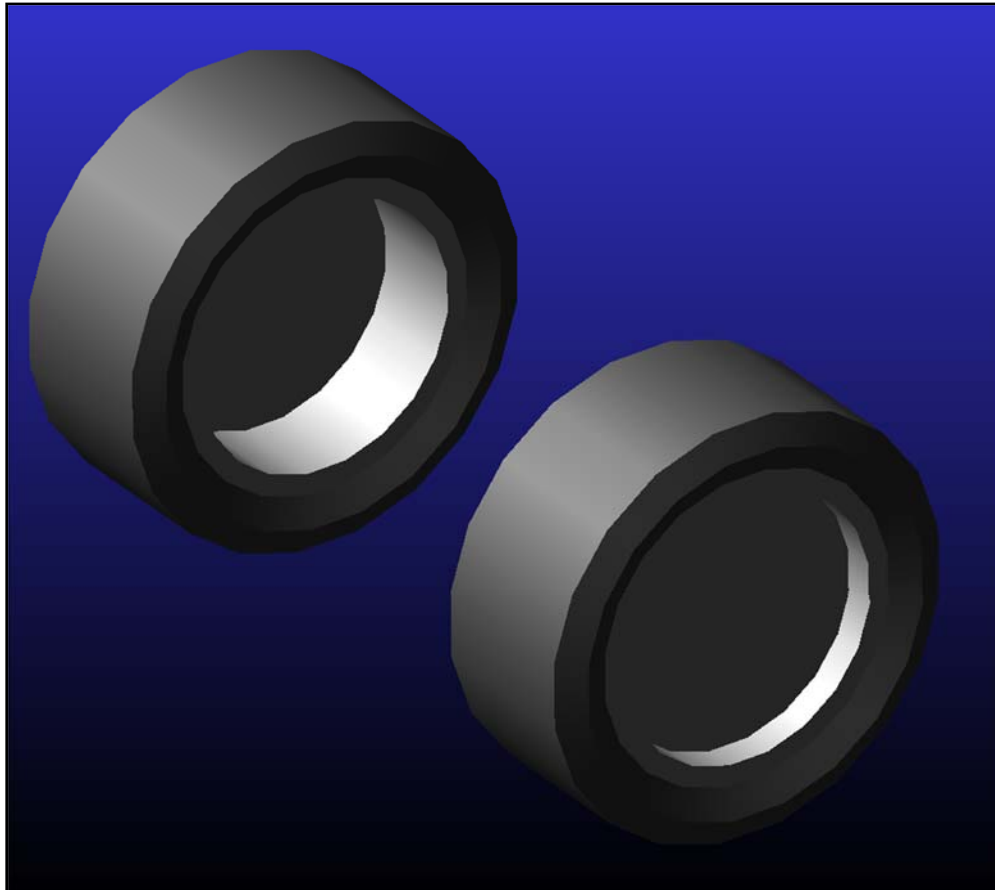
# EXERCISE

- **Perform Workshop 7, “CREATING EVENT FILES”.**



# SECTION 8

## TIRES



# TIRES

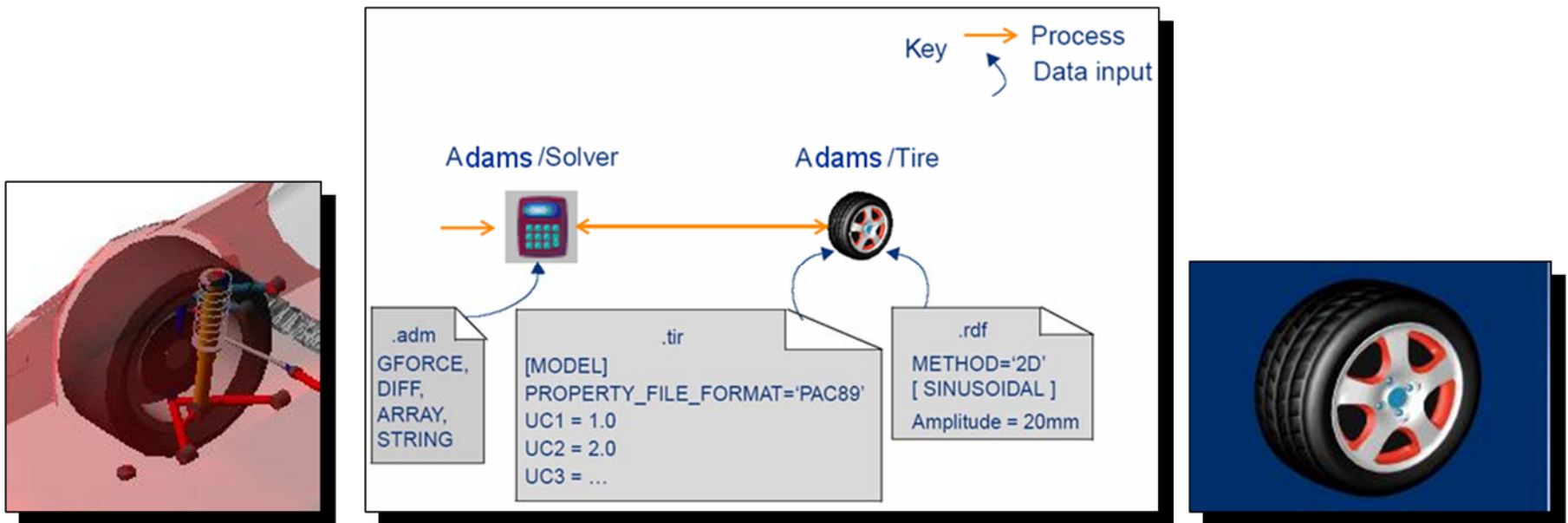
- **This section provides an overview of the calculation of tire forces and the available models.**

# TIRES

- **What's in this section:**
  - Tire Overview
  - Adams/Tire Modules
  - Overview of the Adams/Tire model features
  - Tire Models
  - How You Use Adams/Tire

# TIRE OVERVIEW

- **Adams/Tire calculates the forces and moments that tires exert on the vehicle as a result of the interaction between the tires and the road surface.**





# TIRE OVERVIEW

- **You can use Adams/Tire to model tires for either vehicle-handling or vehicle-durability analyses:**
  - **Handling analyses** are useful for studying vehicle dynamic responses to steering, braking, and throttle inputs.
  - **Durability analyses** are useful for generating road load histories and stress and fatigue studies that require component force and acceleration calculation. These studies can help you calculate the effects of road profiles such as pothole, curb, or Belgian block.

# HANDLING TIRE MODELS

- **The following tire models are available for handling studies:**

- PAC2002 Tire Model\*
- PAC-TIME Tire Model\*
- Pacejka '89 and '94 Models\*
- Fiala Handling Force Model
- UA-Tire Model
- 521-Tire Model

\* The formulae used in the Pacejka tire models are derived from publications by Dr. H.B. Pacejka, and are commonly referred to as the Pacejka method in the automotive industry. Dr. Pacejka himself is not personally associated with the development of these tire models, nor does he endorse them in any way.

# ROAD TYPES

- **Road descriptions vary from simple to complex. The following road models are available:**
  1. 2D Road
    - The 2D Road supports the description of roads with a range of two-dimensional obstacles.
  2. 3D Spline Road
    - 3D Spline Road lets you define an arbitrary three-dimensional smooth road surface. In addition, you can superimpose three-dimensional road obstacles, such as a curb, pothole, ramp, or road crown, on top of the underlying smooth road surface.
  3. 3D Shell Road
    - 3D Shell Road describes a road surface by triangular meshed elements. The road can work with the default 3D Equivalent Volume Contact method or with the 3D Enveloping Contact.

# ROAD TYPES

## 4. OpenCRG Road

- The OpenCRG Road is based on the open source road from the OpenCRG® organization (<http://www.opencrg.org>).
- OpenCRG provides a series of open file formats and tools for the detailed description of (measured) road surfaces.

## 5. Regular Grid Road

- RGR road has been developed by cosin scientific software (<https://www.cosin.eu/>) and is dedicated to measured 3D road surfaces similar to OpenCRG road.
- Also tools are provided for generating, modifying and visualizing this road format.

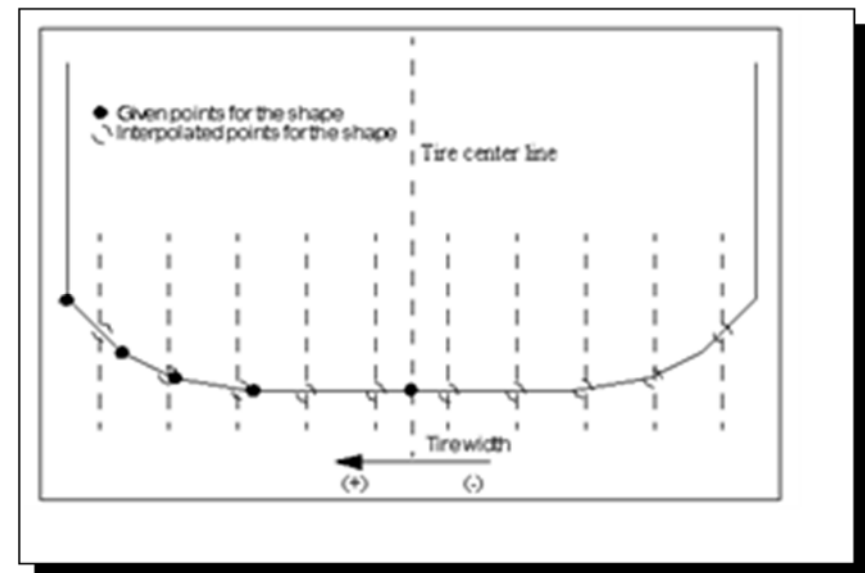
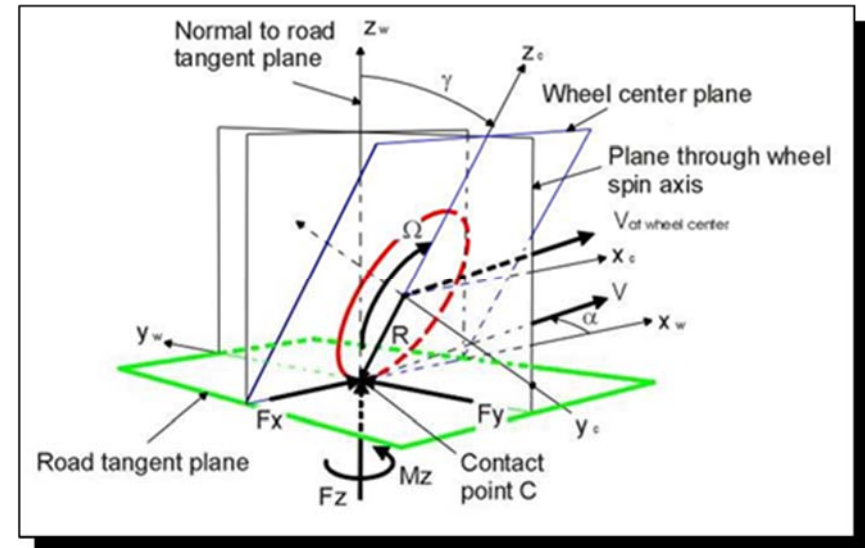
# TIRE-ROAD CONTACT METHODS

## 1. One Point Follower Contact

- The one point follower contact is the default contact method for all tire models (except FTire) in combination with 2D Road, 3D Spline Road, OpenCRG Road and RGR Road.

## 2. 3D Equivalent Volume Contact

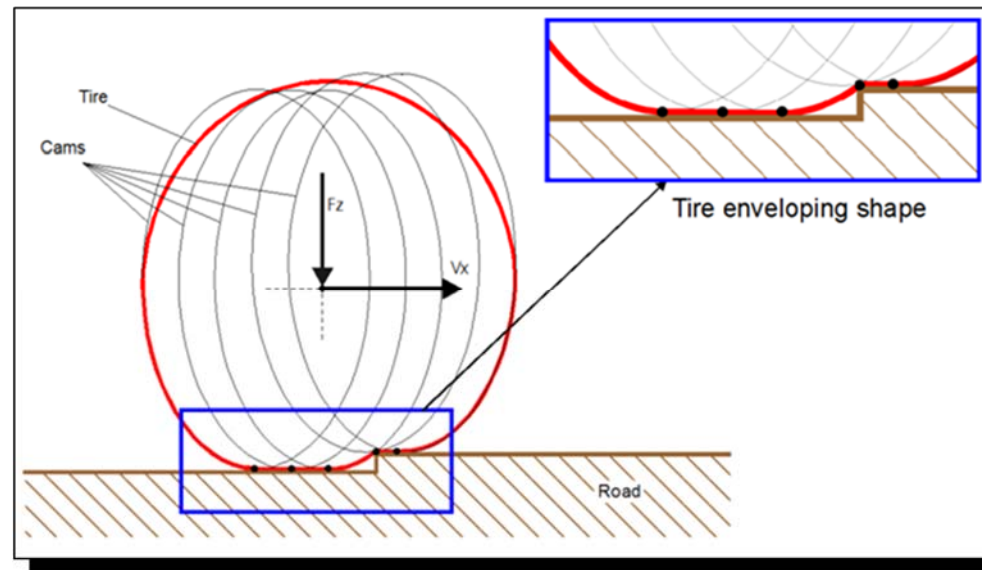
- The 3D Equivalent Volume Contact method can be used with 3D Shell Roads only.
- The 3D Shell Road utilizes a three-dimensional tire-to-road contact model that computes the volume of intersection between the road and the tire.



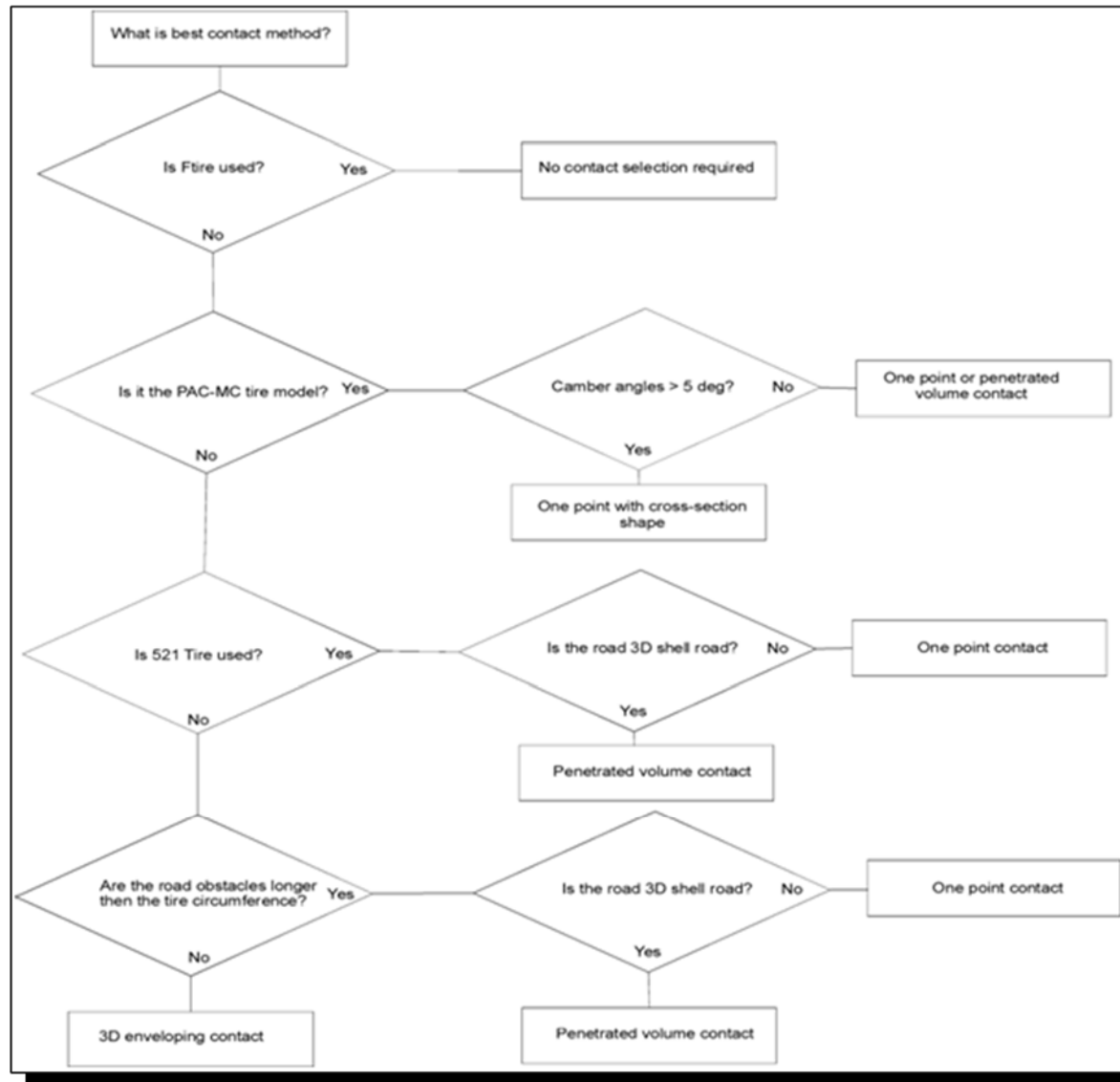
# TIRE-ROAD CONTACT METHODS

## 3. 3D Enveloping Contact

- When driving over road obstacles that have wavelengths shorter than the tire circumference, the tire enveloping behavior starts to play an important role.
- For modeling this non-linear obstacle swallowing effect the so-called 'tandem-egg' or 'tandem-cam' approach is used.
- In the 'tandem egg' or 'tandem cam' approach a series of connected elliptical cams are defined which shape corresponds to the outside tire contour in the tire contact patch zone.



# WHICH CONTACT METHOD SHOULD YOU USE?



# SPECIAL TIRE MODELS

## 1. Pacejka Motorcycle Tire Model

- A Pacejka tire model suitable for motorcycle handling analysis. It describes the tire-road interaction forces with tire-road inclination angles up to 60 degrees.

## 2. Soft Soil Tire Model

- A tire model that can predict the tire-road interaction forces on soft soil road surfaces existing Adams/Tire Road descriptions.

## 3. Aircraft Tire Models

- Tire models that have been developed for aircraft tire modeling. However the Aircraft Basic and Enhanced tire model can be used for other tire types as well.

## 4. Adams/Tire FTire

- FTire can describe the 3D tire dynamic response up to 120 Hz and beyond, due to its flexible ring approach for the tire belt. It can handle any road obstacle.
- In addition to FTire support of all Adams/Tire Road modules, it can deal with the OpenCRG road and specific FTire road formats.



# ADAMS/TIRE TOOLS

## 1. Adams/Tire Test Rig

- The Tire Test Rig application offers the possibility to analyze specific tire responses in a test rig with a single tire under any conditions with any excitations.
- Tire load or wheel axle height can be defined, as well as the tire forward velocity, rotational velocity, steering and camber angle.

## 2. The Adams/Tire Road Builder

- The Road Builder lets you Create, modifying or visualizing 3D Spline Road property files in XML format

## 3. The PAC2002 Tire Data and Fitting Tool (TDFT)

- The PAC2002 Tire Data and Fitting Tool (TDFT) calculates PAC2002 tire model parameters out of tire measurement data for steady-state pure and combined slip conditions including the possibility to visualize/modify tire characteristics.
- In addition the tool can convert existing Adams/Tire property files to a PAC2002 tire property file by fitting on-line generated virtual tire test data.

*Note: Road builder has been detailed in section 9 and TDFT in Appendix E*

# WHICH TIRE MODEL SHOULD YOU USE?

- Each tire model is intended for a certain use case. Using the wrong tire model can result in nonrealistic analysis results.
- The Handling Tire models can describe the first-order response of a tire, but do not take the eigen frequencies of the tire itself into account.
- Therefore, the Handling Tire models are valid up to approximately 8 Hz. PAC2002 can also use a more advanced transient method that extends the validity range up to 15 Hz.
- PAC2002 offers a basic belt dynamics approach(rigid ring part) to increase validity up to 70 - 80 Hz
- FTire is using a more complex approach with a more detailed contact patch and a flexible ring approach, which allows simulations to higher frequencies and a wider range of applications.

# WHICH TIRE MODEL SHOULD YOU USE?

Adams	Event / Maneuver	ADAMS/ Handling Tire							
		PAC2002	PAC2002*	PAC-TIME	PAC89	PAC94	FIALA	5.2.1.	UA Tire
Handling	Stand still and start	+	+	o/+	o/+	o/+	o/+	o/+	o/+
	Parking (standing steering effort)	+	+	-	-	-	-	-	-
	Standing on tilt table	+	+	+	+	+	+	+	+
	Steady state cornering	+	+	+	o/+	+	o	o	o/+
	Lane change	+	+	+	o/+	+	o	o	o/+
	ABS braking distance	+	+	o/+	o/+	o/+	o	o	o/+
	Braking/power-off in a turn	+	+	+	o	o	o	o	o
	Vehicle Roll-over	+	+	o	o	o	o	o	o
	On-line scaling tire properties	+	+	-	-	-	-	-	-
Ride	Cornering on uneven roads <sup>1</sup>	o/+	+	o	o	o	o	o	o
	Braking on uneven roads <sup>1</sup>	o/+	+	o	o	o	o	o	o
	Crossing cleats / obstacles	-	+	-	-	-	-	-	-
	Driving over uneven road	-	+	-	-	-	-	-	-
	4 post rig (A/Ride)	+	+	o/+	o/+	o/+	o/+	o/+	o/+
Chassis Control	ABS braking control	o/+	+	o	o	o	o	o	o
	Shimmy <sup>2</sup>	o/+	+	o	o	o	o	o	o
	Steering system vibrations	o/+	+	o	o	o	o	o	o
	Real-time	+	-	-	-	-	-	-	-
	Chassis control systems > 8 Hz	o/+	+	-	-	-	-	-	-
	Chassis control with ride	-	+	-	-	-	-	-	-
Dura-bility	Driving over curb	-	o/+	-	-	-	-	o	o
	Driving over curb with rim impact	o	o/+	-	-	-	-	o	o
	Passing pothole	-	o/+	-	-	-	-	o	o
	Load cases	-	o/+	-	-	-	-	o	o
Misc	Design of Experiments	-	+	-	-	-	-	-	-
	SMP parallel	+	+	+	+	+	+	+	+

-

Not possible/Not realistic

o

Possible

o/+

Better

+

Best to use

<sup>1</sup>wavelength road obstacles > tire diameter  
<sup>2</sup>wheel yawing vibration due to suspension flexibility and tire dynamic response  
 tire models assumed to be used in transient and combined slip mode  
 \* PAC2002 with belt dynamics

# WHICH TIRE MODEL SHOULD YOU USE?

Adams	Event / Maneuver	Specific Models			Aircraft		
		PAC-MC	FTire	SoftSoil	Basic	Enhanced	TRR64
Handling	Stand still and start	o/+	+	-	o/+	o/+	o/+
	Parking (standing steering effort)	-	+	-	-	-	-
	Standing on tilt table	+	+	-	+	+	+
	Steady state cornering	+	o/+	+	o	+	o/+
	Lane change	+	o/+	+	o	+	o/+
	ABS braking distance	o/+	+	o/+	o	o/+	o/+
	Braking/power-off in a turn	+	o/+	+	o	o	o
	Vehicle Roll-over	o	+	o/+	o	o	o
	On-line scaling tire properties	-	o	-	-	-	-
Ride	Cornering on uneven roads <sup>1</sup>	o	+	o	o	o	o
	Braking on uneven roads <sup>1</sup>	o	+	o	o	o	o
	Crossing cleats / obstacles	-	+	o	-	-	-
	Driving over uneven road	-	+	o	-	-	-
	4 post rig (A/Ride)	o/+	+	-	o/+	o/+	o/+
Chassis Control	ABS braking control	o	+	-	o	o	o
	Shimmy <sup>2</sup>	o	+	-	o	o	o
	Steering system vibrations	o	+	-	o	o	o
	Real-time	-	-	-	-	-	-
	Chassis control systems > 8 Hz	-	+	o	-	-	-
	Chassis control with ride	-	+	-	-	-	-
Dura-bility	Driving over curb	-	+	o	-	o	o
	Driving over curb with rim impact	-	+	-	-	o	o
	Passing pothole	-	+	o	-	o	o
	Load cases	-	+	o	-	o	o
Misc	Design of Experiments	-	+	-	-	-	-
	SMP parallel	+	+	+	+	+	+

-

Not possible/Not realistic

o

Possible

o/+

Better

+

Best to use

<sup>1</sup> wavelength road obstacles > tire diameter  
<sup>2</sup> wheel yawing vibration due to  
suspension flexibility and tire dynamic response  
tire models assumed to be used in transient and combined slip mode



# OVERVIEW OF THE ADAMS/TIRE MODEL FEATURES

- All the tire models can deal with all Adams road types, however not all contact methods can be used with all road and tire model combinations.
- In general the One Point Follower, the 3D Equivalent Volume Contact and the Tire Cross Section Profile Contact are valid for rather smooth roads only
- The 3D Enveloping Contact can deal with any shape and size of road obstacles at all road types (2D Road, 3D Spline, 3D meshed, OpenCRG Road and RGR Road).

tire model features		PAC89	PAC94	PAC2002	PAC-MC	Fiala	UA-Tire	S21 Tire	Air Basic	Air Enhanced	Air TRR64	Soft Soil
contact methods	one-point contact (default)	X	X	X	X	X	X	X	X	X	X	X
	penetrated volume contact	X	X	X	X	X	X	X	X	X	X	X
	3D enveloping contact	X	X	X		X	X		X	X	X	
	tire cross-section profile contact				X							
transient	linear transient (relaxation lengths)	X	X	X	X	X	X	X	X	X	X	X
	non-linear (advanced) transient			X								
belt dynamics				X								
design of experiments				X								
non-linear vertical stiffness		X	X	X	X	X	X	X	X	X	X	X
rim-road contact				X					X	X	X	X
advanced loaded radius modeling				X								
parking torque				X								
frequency dependent stiffness (standing tire)				X								
conversion to PAC2002 with TDFT		X	X		X	X	X		X	X	X	
on-line scaling				X								
SmartDriver-Tire interface support		X	X	X	X	X	X	X	X	X	X	X

# HOW YOU USE ADAMS/TIRE

- **Tire models are specified in the tire property files. Adams/Solver chooses the corresponding tire formulation based on the property file information. Example:**

```
[MODEL]
PROPERTY_FILE_FORMAT    ='PAC2002'
USE_MODE                = 14                $Tyre use switch (IUSED)
VXLOW                   = 1
LONGVL                  = 16.6              $Measurement speed
TYRESIDE                 = 'LEFT'           $Mounted side of tyre at vehicle/test bench
```

# TIRE USE\_MODE

- The **USE\_MODE** specifies the complexity of the tire model. For example, for PAC2002 tire models you can specify to calculate Fz only, Fx and My only or you can have all the tire forces calculated:

```
USE_MODE specifies the type of calculation performed:  
  0: Fz only, no Magic Formula evaluation  
  1: Fx, My only  
  2: Fy, Mx, Mz only  
  3: Fx, Fy, Mx, My, Mz uncombined force/moment calculation  
  4: Fx, Fy, Mx, My, Mz combined force/moment calculation  
+10: including relaxation behaviour  
*-1: mirroring of tyre characteristics  
  
example: USE_MODE = -12 implies:  
  -calculation of Fy, Mx, Mz only  
  -including relaxation effects  
  -mirrored tyre characteristics
```

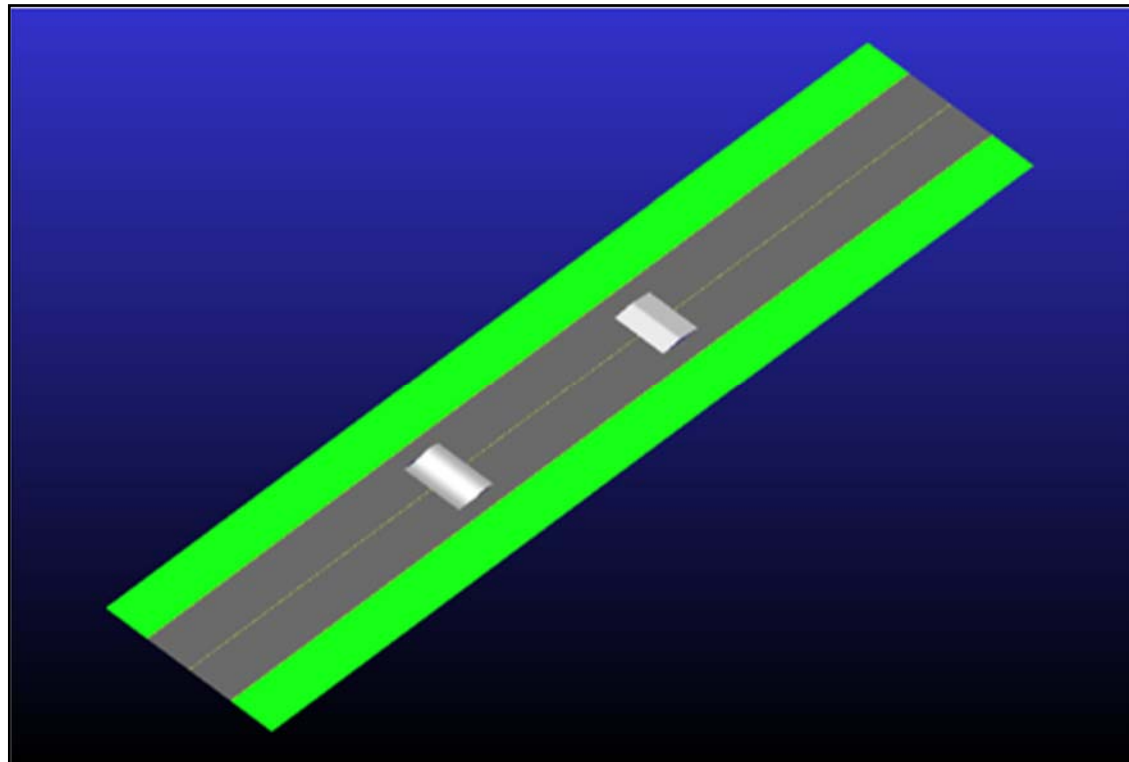
# EXERCISE

- **Perform Workshop 8, “TIRE TESTRIG TUTORIAL”.**



# SECTION 9

## ROAD BUILDER



# ROAD BUILDER

- **Adams/Car supports a number of different road representations.**
- **The Road Builder lets you create and edit 3D-Spline Road property files in XML format.**

# ROAD BUILDER

- **What's in this module:**
  - Road Types
  - Why use Road Builder?
  - Using the Road Builder
    - Starting Road Builder
    - Setting Global Parameters
    - Defining Road Data Points
    - Plotting Road Data Points
    - Defining Obstacles
    - Road Generator
    - Defining Analytical Roads

# ROAD TYPES

- **2D Road Model**
- **3D Spline Road Model**
- **3D Shell Road Model**
- **Soft Soil Road Model**
- **OpenCRG Road Model**
- **Regular Grid Road (RGR) Model**

# ROAD TYPES

- **2D Road Model**

- The 2D road contact uses a point-follower method (for example, think of a disk of a plane).
- The following are the different road types that work with this method (primarily for FTIRE):
  - **DRUM** - Tire test drum (requires a zero-speed-capable tire model).
  - **FLAT** - Flat road.
  - **PLANK** - Single plank perpendicular, or in oblique direction relative to x-axis, with or without bevel edges.
  - **POLY\_LINE** - Piece-wise linear description of the road profile. The profiles for the left and right track are independent.
  - **POT\_HOLE** - Single pothole of rectangular shape.
  - **RAMP** - Single ramp, either rising or falling.
  - **ROOF** - Single roof-shaped, triangular obstacle.

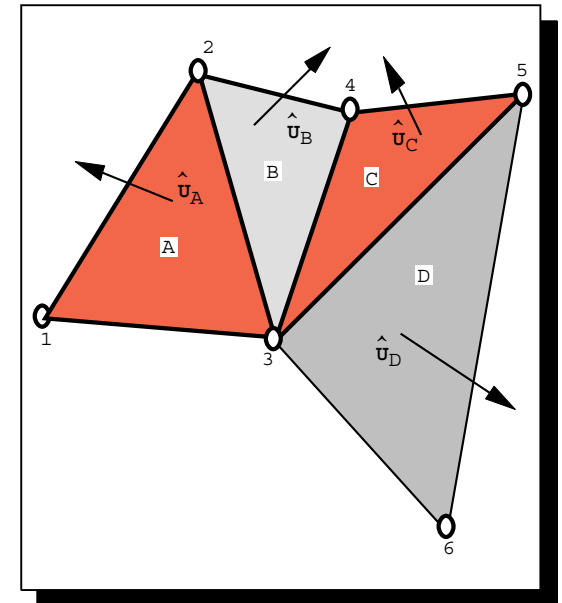
# ROAD TYPES

- **2D Road Model (Cont.)**
  - **SINE** - Sine waves with constant wave length.
  - **SINE\_SWEEP** - Sine waves with decreasing wave lengths.
  - **STOCHASTIC\_UNEVEN** - Synthetically generated irregular road profiles that match measured stochastic properties of typical roads. The profiles for left and right track are independent, or may have a certain correlation.
- **Sample files for all the road types for Adams/Car are in the standard Adams/Car database:**
  - `<install_dir>/acar/shared_car_database.cdb/roads.tbl/`

# ROAD TYPES

- **3D Shell Road Model**

- The 3D Shell Road model is a three-dimensional tire-to-road contact model (3D Equivalent Volume Contact) that **computes the volume of intersection between a road and tire**.
- The road is modeled as a set of discrete triangular patches and the tire as a set of cylinders.
- About the 3D equivalent volume road model
  - The figure depicts a road surface formed by six nodes numbered 1 through 6. The six nodes together form four triangular patch elements denoted as A, B, C, and D. The unit outward normal for each triangular patch is shown for the sake of clarity.
  - Much like finite-element mesh convention, Adams/Tire requires that you define a road by first specifying the coordinates of each node in the road reference-marker axis system. Subsequently, you specify the three nodes that form each triangular patch.



# ROAD TYPES

- **3D Shell Road Model (Cont.)**
- **Defining the 3D shell road surface**
  - You use a road property file to define the three-dimensional road surface. The road property file consists of five data blocks: Header, Units, Model, Nodes, and Elements.
  - These blocks of data can appear in any order in the file, and keywords can appear in any order within the block to which they belong.

```
$-----MDI_HEADER
[MDI_HEADER]
FILE_TYPE = 'rdf'
FILE_VERSION = 5.00
FILE_FORMAT = 'ASCII'
(COMMENTS)
{comment_string}
'Big bump for testing durability tire.'
$-----units
[UNITS]
LENGTH = 'mm'
FORCE = 'newton'
ANGLE = 'radians'
MASS = 'kg'
TIME = 'sec'
$-----definition
[MODEL]
METHOD = '3D'
$-----refsys
[REFSYS]
OFFSET = 0.0 0.0 0.0
ROTATION_ANGLE_XY_PLANE = 0.0
$-----nodes
[NODES]
NUMBER_OF_NODES = 10
{ node x_value y_value z_value }
1 100000.0 10000.0 0.0
2 -24000.0 10000.0 0.0
3 -24000.0 10000.0 100
4 -100000.0 10000.0 100
5 -150000.0 10000.0 100
6 100000.0 -10000.0 0.0
7 -17000.0 -10000.0 0.0
8 -17000.0 -10000.0 100
9 -100000.0 -10000.0 100
10 -150000.0 -10000.0 100
$-----elements
[ELEMENTS]
NUMBER_OF_ELEMENTS = 8
{ node_1 node_2 node_3 mu }
1 7 6 1.0
1 2 7 1.0
```



# ROAD TYPES

- **3D Spline Road:**

- The 3D Spline road option lets you model many types of three-dimensional smooth roads, such as parking structures, race tracks, and so on.
- Information can be stored in either **TeimOrbit** or **XML** format.
- The example shows a 3D Spline road file in TeimOrbit (.rdf) format.
- The Road Builder Utility makes it **easy to build 3D Spline road models and stores them in the XML format.**

```
$-----MDI_HEADER
[MDI_HEADER]
FILE_TYPE = 'rdf'
FILE_VERSION = 5.00
FILE_FORMAT = 'ASCII'
(COMMENTS)
{comment_string}
'3d smooth road'
$-----UNITS
[UNITS]
LENGTH = 'meter'
FORCE = 'newton'
ANGLE = 'radians'
MASS = 'kg'
TIME = 'sec'
$-----DEFINITION
[MODEL]
METHOD = '3D_SPLINE'
FUNCTION_NAME = 'ARC903'
VERSION = 1.00
$-----ROAD_PARAMETERS
[GLOBAL_PARAMETERS]
CLOSED_ROAD = 'no'
SEARCH_ALGORITHM = 'FaSt'
ROAD_VERTICAL = '0.0 0.0 1.0'
FORWARD_DIR = 'NORMAL'
MU_LEFT = 1.0
MU_RIGHT = 1.0
WIDTH = 7.000
BANK = 0.0
$-----DATA_POINTS
[DATA_POINTS]
{ X Y Z WIDTH BANK MU_LEFT MU_RIGHT }
12.50000E+00 0.00000E-00 0.00000E-00 7.000 0.000 0.900 0.900
10.50000E+00 0.00000E-00 0.00000E-00 7.000 0.000 0.900 0.900
5.50000E+00 0.00000E-00 0.00000E-00 7.000 0.000 0.900 0.900
0.50000E+00 0.00000E-00 0.00000E-00 7.000 0.000 0.900 0.900
0.00000E+00 0.00000E-00 0.00000E-00 7.000 0.000 0.900 0.900
-2.50000E+00 0.00000E-00 0.00000E-00 7.000 0.000 0.900 0.900
-5.00000E+00 0.00000E-00 0.00000E-00 7.000 0.000 0.900 0.900
-1.00000E+01 0.00000E-00 0.00000E-00 7.000 0.000 0.900 0.900
-2.00000E+01 0.00000E-00 0.10000E-00 7.000 0.000 0.900 0.900
-3.00000E+01 0.00000E-00 0.20000E-00 7.000 0.000 0.900 0.900
-4.00000E+01 0.00000E-00 0.30000E-00 7.000 0.000 0.900 0.900
-5.00000E+01 0.00000E-00 0.40000E-00 7.000 0.000 0.900 0.900
-6.00000E+01 0.00000E-00 0.50000E-00 7.000 0.000 0.900 0.900
-7.00000E+01 0.00000E-00 0.60000E-00 7.000 0.000 0.900 0.900
-8.00000E+01 0.00000E-00 0.70000E-00 7.000 0.000 0.900 0.900
-9.00000E+01 0.00000E-00 0.80000E-00 7.000 0.000 0.900 0.900
-1.00000E+02 0.00000E-00 0.90000E-00 7.000 0.000 0.900 0.900
-1.10000E+02 0.00000E-00 1.00000E+00 7.000 0.000 0.900 0.900
-1.20000E+02 0.00000E-00 1.10000E+00 7.000 0.000 0.900 0.900
-1.30000E+02 0.00000E-00 1.20000E+00 7.000 0.000 0.900 0.900
$-----END_DATA_POINTS
```

# ROAD TYPES

- **3D Spline Road (Cont.):**

- 3D Spline Road can be used in the following ways:

- Full vehicle simulations can use an Adams/3D Spline Road data file for the road.
- Both Adams/SmartDriver and the Driving Machine can accept path information from 3D Spline Road files.

# ROAD TYPES

- **Soft Soil Road Model**

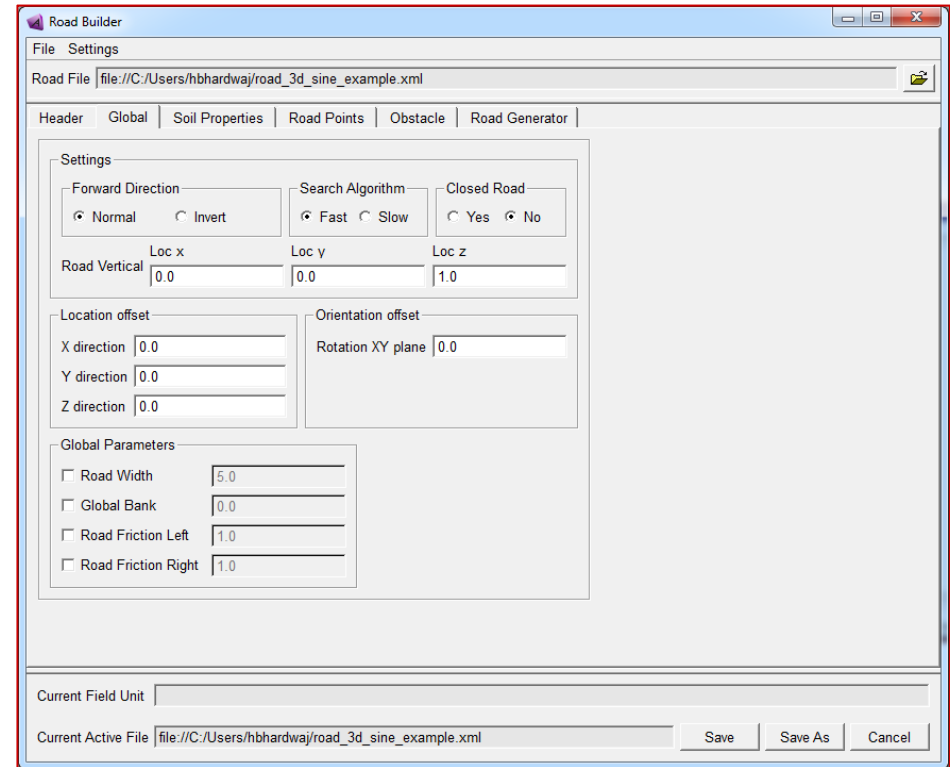
- The Adams/Tire Soft Soil tire model offers a basic model to describe the **tire-soil interaction forces** for any tire on elastic/plastic grounds, such as sand, clay, loam and snow.
- The tire model uses state-of-the-art knowledge about terra-mechanical tire-soil interaction as published by Bekker, Wong, Janosi, Ishigami and Schmid.
- The application range covers the basic driving maneuvers with **‘one-point’ of contact tire-road methods** in which no excessive camber occurs.
- Existing Adams/Tire road formats can be used, with a few soil property parameters added: either **an elastic-plastic approach or an elastic-viscous approach** can be chosen for the soil modeling.

# WHY USE ROAD BUILDER?

- **Convenient tool for creating and editing 3D Spline road data file**
- **Using Road Builder you can**
  - Use a plot or table to edit the road points
  - Visualize the road
  - Add soil properties
  - Create variety of road obstacles to customize your testing track

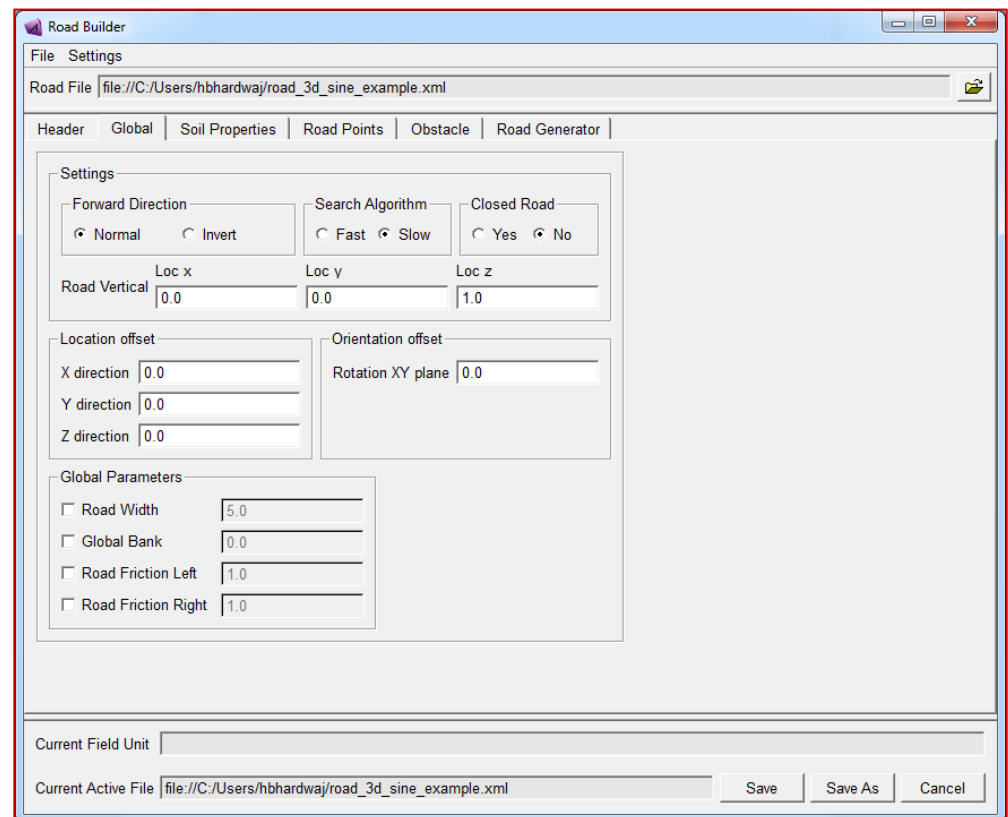
# USING THE ROAD BUILDER

- **Starting the Road Builder:**
  - To start the Road Builder in Adams/Car:
    - From the Simulate menu, point to Full-Vehicle Analysis, and then select Road Builder
  - The Road Builder consists of five tabs:
    - Header
    - Global
    - Road Points
    - Obstacle
    - Road Generator



# USING THE ROAD BUILDER

- **To set or edit the global parameters:**
  - Select the **Global** tab.
  - Change the parameters. Refer to the online help for detail description on parameters, Forward Directions, Road Vertical, Search Algorithm, Mu\_Left, Mu\_Light, Width and Bank



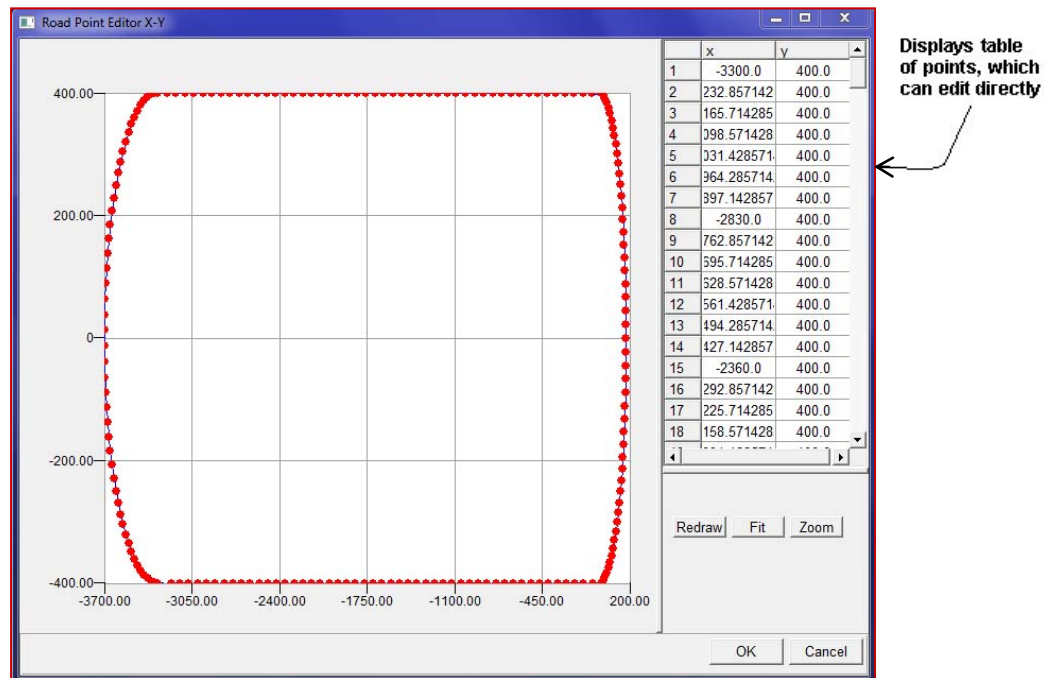
Displays units of currently  
selected parameter



# USING THE ROAD BUILDER

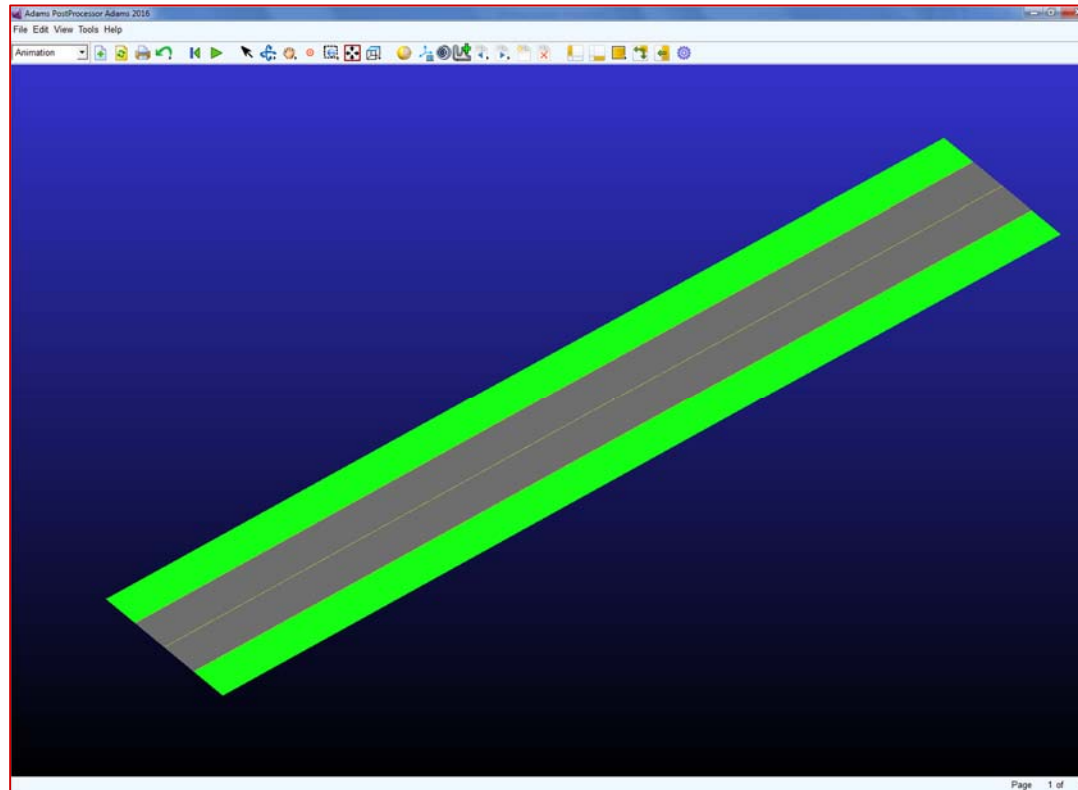
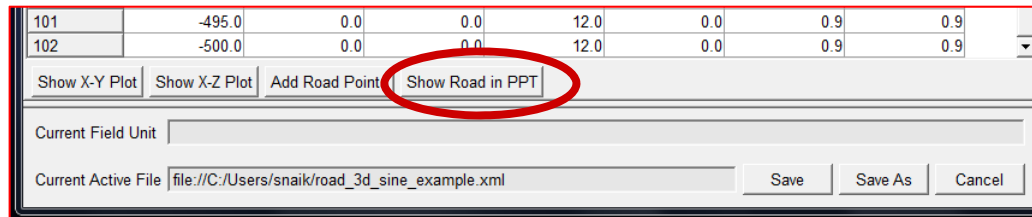
- **Plotting Road Data Points:**

- You can visualize the road data plots by plotting them as x-y (x values versus y values) or x-z plots (x values versus z values).
- Fit the display, zoom the display
- Modify the road data points
- Generate 3D Road



# USING THE ROAD BUILDER

- **Generate 3D Road**





# USING THE ROAD BUILDER

- **Defining Obstacles:**

- Obstacle Types
- Common Parameters
- Obstacle Specific Parameters

The screenshot shows the 'Road Builder' software window with the 'Obstacle' tab selected. The 'Road File' is set to 'mdids://acar\_shared/roads.tbl/3d\_road\_obstacle\_sine.xml'. The 'Name' field is 'SINE' and the 'Comment' field is empty. Under 'Common Parameters', the 'Obstacle Type' is 'sine', 'Width' is '2.0', 'Length' is '80.0', and 'Friction' is '0.9'. The 'Coordinate System' is set to 'Local'. The 'Start Location' and 'Stop Location' are defined by 'Loc X', 'Loc Y', and 'Loc Z' coordinates. For 'Start Location', the values are -60.0, 0.0, and 0.0 respectively. For 'Stop Location', the values are -70.0, 0.0, and 0.0 respectively. Under 'Obstacle Specific Parameters', 'Amplitude' is '0.2', 'Wavelength Offset' is '0.0', and 'Wavelength' is '5.0'. The 'Current Field Unit' is 'length (meter)'. At the bottom, the 'Current Active File' is the same as the 'Road File', with 'Save', 'Save As', and 'Cancel' buttons.

Loc X			Loc Y			Loc Z		
Start Location	-60.0		0.0		0.0			
Stop Location	-70.0		0.0		0.0			

# USING THE ROAD BUILDER

- Road Generator

The screenshot shows the 'Road Builder' application window with the 'Road Generator' tab selected. The interface includes a menu bar (File, Settings), a 'Road File' path, and a tabbed navigation system. The 'Road Generator' tab displays a table with road segments and their properties.

Name	Type	Number of Points	Start Point	Center Point	Tangent Point	End Point	Radius	Arc Length	Width
segment_1	Linear	50	0.0, 400.0, 0.0	N/A	N/A	0.0, 0.0, 0.0	N/A	N/A	N/A
segment_2	Transition	50	N/A	N/A	N/A	N/A	N/A	N/A	N/A
segment_3	Linear	50	0.0, 400.0, 0.0	N/A	N/A	0.0, 0.0, 0.0	N/A	N/A	N/A
segment_4	Curvature	50	0.0, 400.0, 0.0	0.0, 0.0, 0.0	500.0, 400.0, 0.0	N/A	400.0	1256.0	N/A
segment_5	User Points	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Below the table, there is a scrollable area and a section for adding new segments. The 'Add' button is visible, along with input fields for 'Name' (segment\_5), 'Type' (RoadSegment), and 'Parent' (Road3D). There is also an 'Export Points to Data Table' button.

At the bottom, the 'Current Field Unit' is set to 'length (meter)', and the 'Current Active File' path is shown. 'Save', 'Save As', and 'Cancel' buttons are located at the bottom right.

# USING THE ROAD BUILDER

- **Defining Analytical Road:**

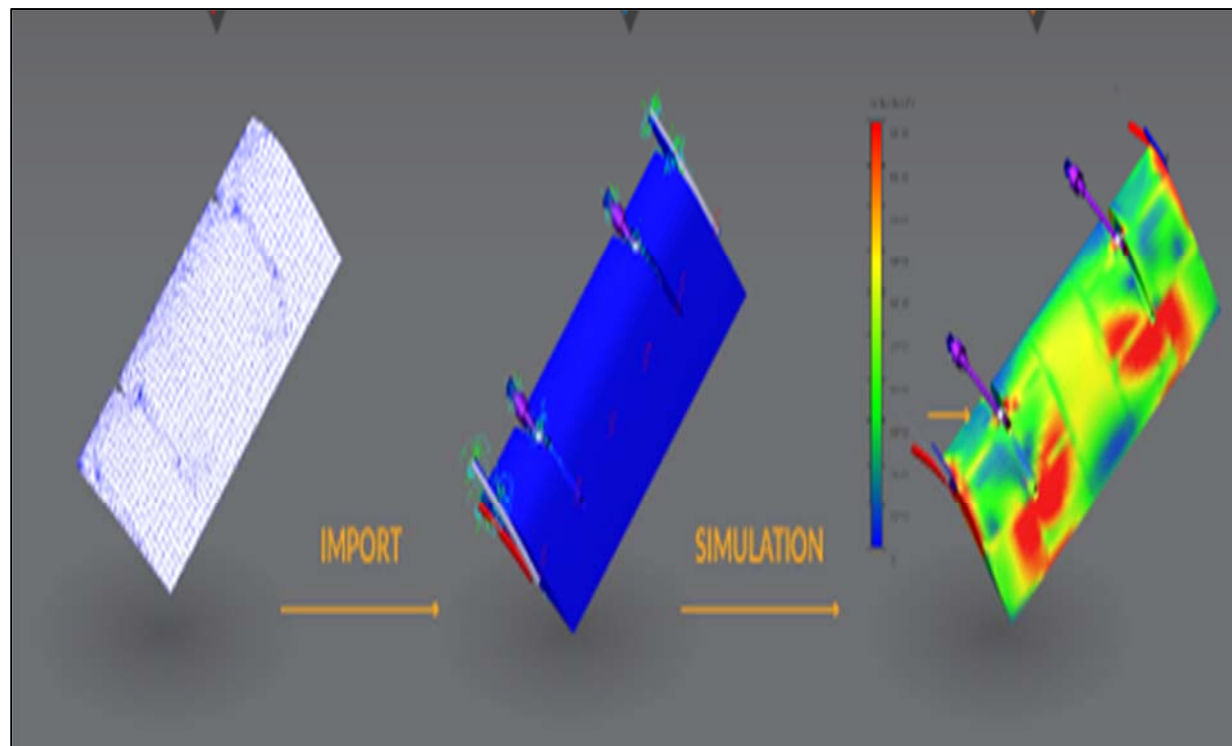
- Road Generator to create/modify a road model analytically from scratch in Adams/Road Builder.
- Road data can be created with multiple segments, each segment representing predefined formulations like
  - Linear, Curvature, and
  - Transition Curve or through
  - User Defined Functions and
  - User Defined Points.
- “Export points to Data Table” – calculates the road points according to the segment function and export them to the Road Points tab in the GUI

# EXERCISE

- **Perform Workshop 9, “CREATING ROAD PROPERTY FILES USING ROAD BUILDER”.**

# SECTION 10

## USING FLEXIBLE BODIES



# USING FLEXIBLE BODIES

- **In this section, you will learn how to create flexible bodies in your models, as well as how to swap a rigid body for a flexible body.**

# USING FLEXIBLE BODIES

- **What's in this section:**
  - Flexible Body Overview
  - Limitations of Flexible Bodies
  - Getting Flexible Bodies
  - Modal Superposition
  - Visualization Attributes
  - About Joints and Motions

# FLEXIBLE BODY OVERVIEW

- **Adams/Flex uses an assumed-modes method of modeling flexible bodies, called modal flexibility.**
- **Modal flexibility assigns a set of mode shapes to a flexible body.**
- **This modal method of modeling flexibility can be very useful in problems that are characterized by high elasticity and moderate deflections. That is, deflections less than 10% of a characteristic length of the body.**



# FLEXIBLE BODY OVERVIEW

- **By integrating flexible bodies into your model, you can:**
  - Capture inertial and compliance properties during handling and comfort simulations.
  - Predict loads with greater accuracy by allowing Adams to account for flexibility during simulations.
  - Study deformation (stress/strain can be calculated using Adams/Durability).
  - Examine the linear system modes of a flexible model when you use Adams/Flex with Adams/Linear.
- **You should use flexible bodies wherever you expect component flexibility to affect the dynamic behavior of your model or when you require accurate information about the deformations of a component in your model.**

# LIMITATIONS OF FLEXIBLE BODIES

- **When you use flexible bodies, remember that flexible body deformations are a linear combination of deformation shapes.**
- **Consequently, take special precautions when modeling higher order deformations, such as those that occur when deformations are large, or when attempting to correctly model centrifugal stiffening of rotating systems.**
- **You can overcome these limitations by dividing a flexible body into multiple flexible bodies and assembling them in Adams/Car.**
- **Also, note that flexible bodies are not parametric. If you want to substitute a new flexible body in your system, you must create a new flexible body.**

# GETTING FLEXIBLE BODIES

- **Two ways to incorporate flexible bodies**
  - Importing modal neutral files (.mnf) - To create a new flexible body, go to **Build → Part → Flexible Body** in the Template Builder. Adams/Car imports the .mnf file and creates the flexible body. You need to create interface part at the interface nodes to attach other parts to the flexible body.
  - Replacing a rigid part with a flexible part at the subsystem level using the Flex Swap Dialog box.



For more information on Adams/Flex, see the Adams/Flex online help.

# MODAL SUPERPOSITION

- **Adams/Flex treats flexible body deformations as small, linear deformations relative to a local reference frame undergoing large motion**
- **Represent deformation as a linear combination of mode shapes**
- **Simple example**

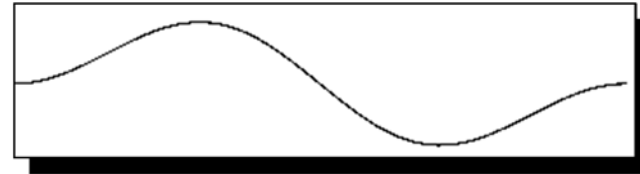
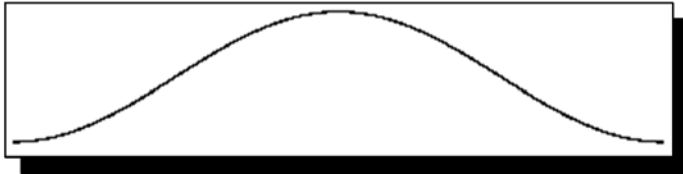
$$\mathbf{u} = \sum_{i=1}^M \phi_i q_i$$

- **Craig-Bampton modes distinguish boundary nodes from interior nodes**

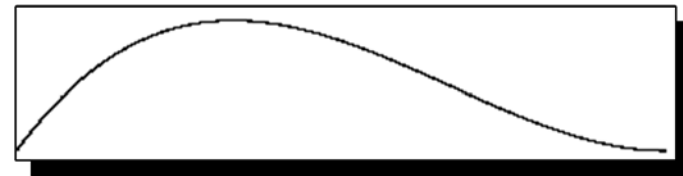
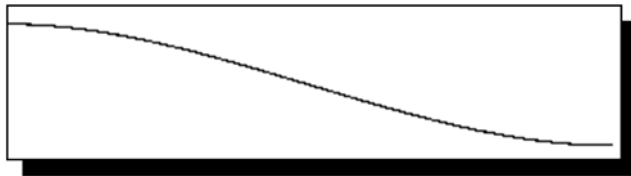


# MODAL SUPERPOSITION

- **Fixed boundary normal modes**



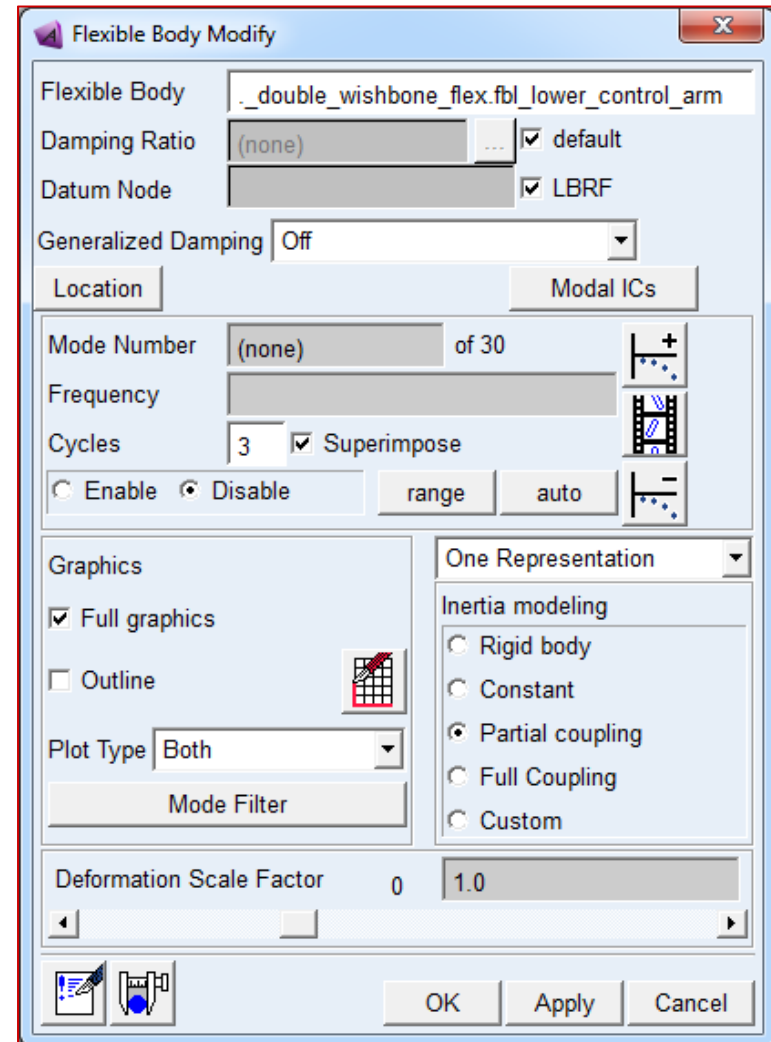
- **Constraint modes**



For more Adams/Flex theory, see the Adams/Flex online help (specifically, see the index entry Craig-Bampton modes - considerations when translating FE models).

# VISUALIZATION ATTRIBUTES

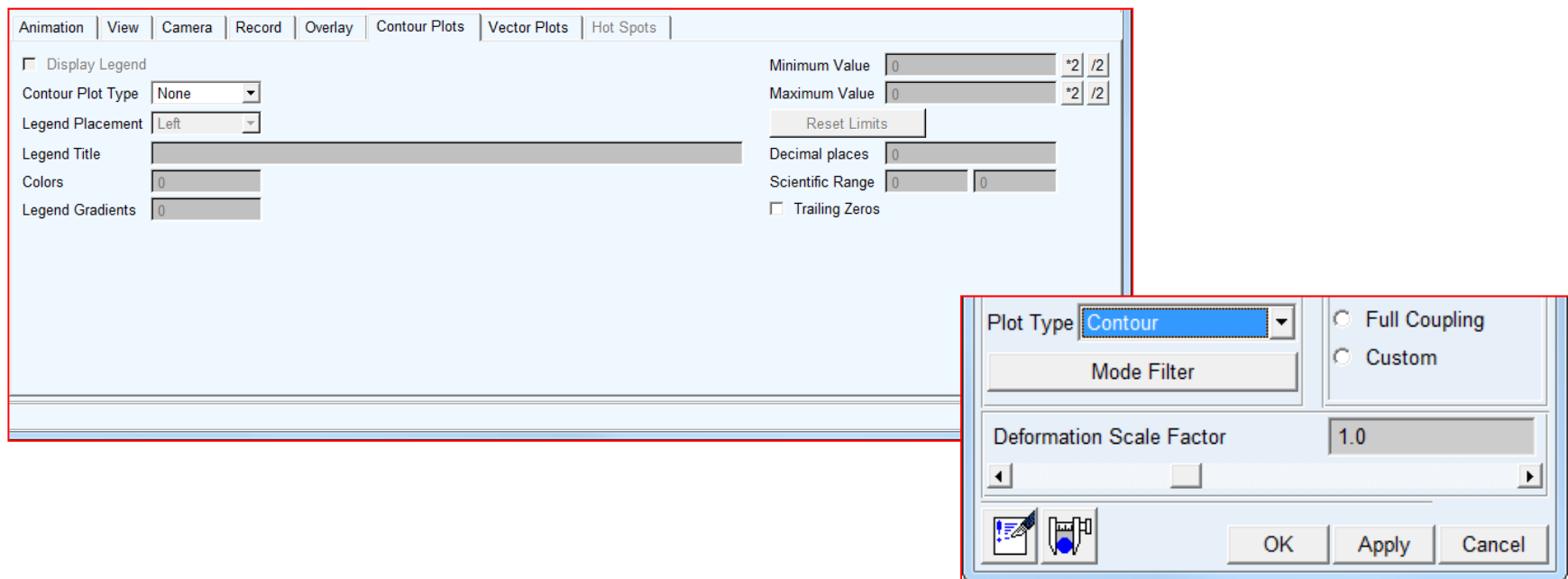
- You can use the Plot Type option to display contour or vector plots.



# VISUALIZATION ATTRIBUTES

- **Color contours**

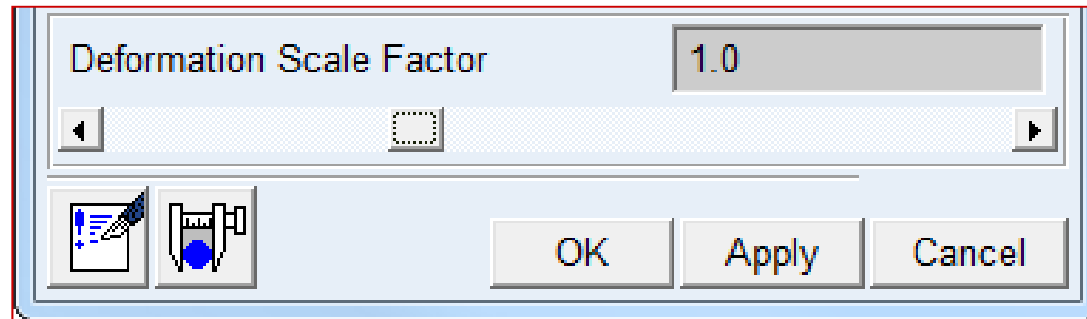
- Indicate the deformation magnitude of the flexible component
- Are continuous attributes, not discrete nodal values
- Show relative deformation, not stress
- Contours turned on in Adams/PostProcessor with the Contour Plots tab



# VISUALIZATION ATTRIBUTES

- **Deformation scale factor**

- Used to exaggerate deformation
- Can scale up or down
- Constraints can appear violated when the scale is  $>1$ . This is merely visual: the analysis will, of course, maintain the constraints you've defined.
- When the scale is equal to 0, the shape will not be deformed but the colors will be displayed.

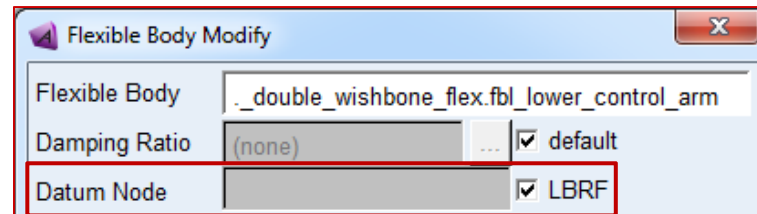




# VISUALIZATION ATTRIBUTES

- **Datum node**

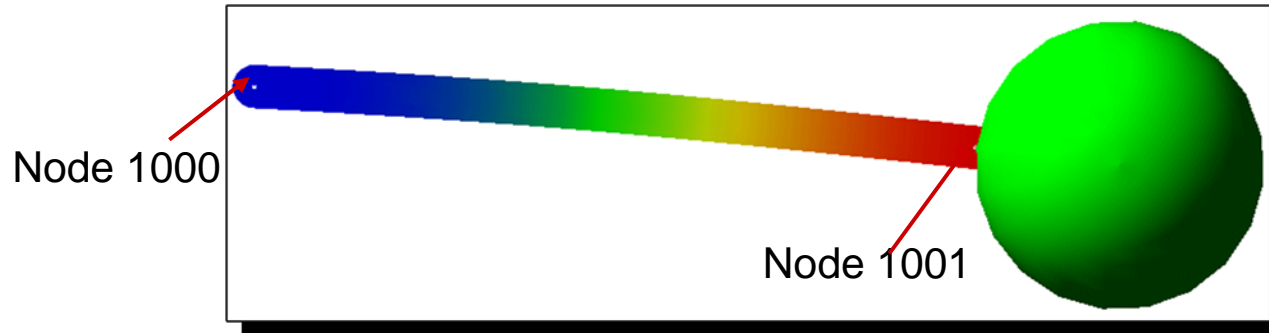
- Deformation is a relative term and should be expressed with respect to a node:
  - You select which node Adams/Flex considers as the undeformed (datum) node and then all other nodes deform relative to it
  - Nodal deformations are color coded relative to the datum node
  - LBRF (local body reference frame, the default, also known as the body coordinate system or BCS) is in the same location as the reference frame used in FEA



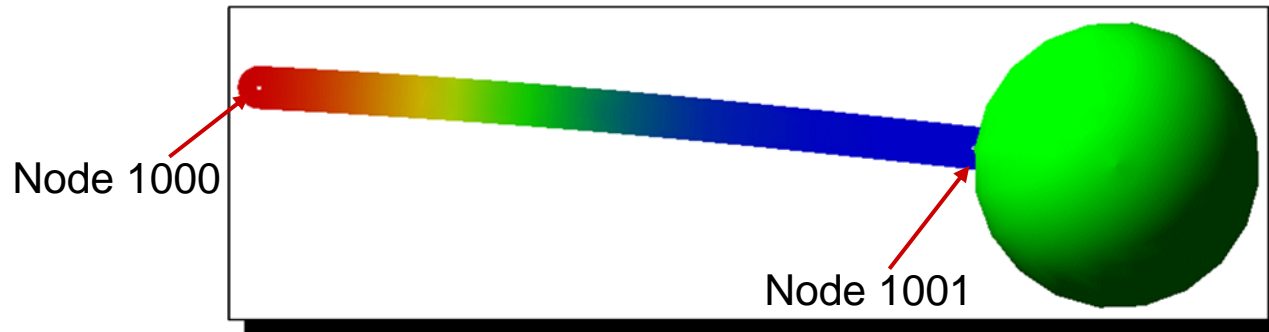
# VISUALIZATION ATTRIBUTES

- **Example of using different datum nodes**

- Here, node 1000 is the datum node.



- In this case, node 1001 is the datum node.



- The color red denotes maximum deformation relative to the datum node.

# ABOUT JOINTS AND MOTIONS

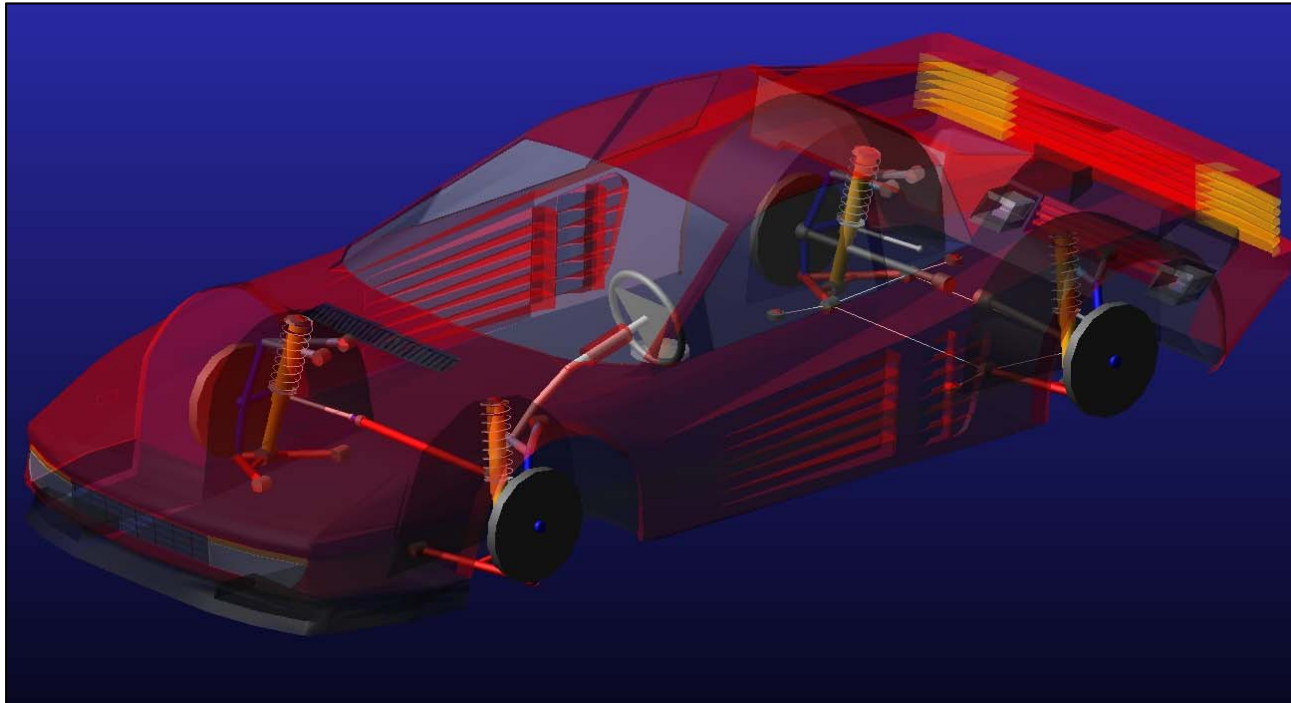
- **Using joints**
  - After you bring a flexible body into Adams/Car, you can connect it to your rigid model using the Adams/Car library of constraints.
- **You can avoid a nodal mismatch by:**
  - Using consistent numbering
  - Paying attention to alignment issues
- **Joints are commonly used to attach flexible bodies**
  - Fixed
  - Revolute
  - Spherical
  - Universal (or Hooke)
- **If you build a template with a flexible body then you need to create interface parts to attach joints to the flex body.**
- **On the other hand if you use the flex swap tool at the subsystem level the additional step is not required.**

# EXERCISE

- **Perform Workshop 10, “FLEX TUTORIAL”**

# SECTION 11

## GENERAL ACTUATION ANALYSIS



# GENERAL ACTUATION ANALYSIS (GAA)

- **In this section you will learn**
  - What General Actuation Analysis (GAA) is
  - How to access actuator parameters
  - Use of the request map editor

# GENERAL ACTUATION ANALYSIS (GAA)

- **What's in this section:**
  - What is GAA?
  - GAA model details
  - Accessing actuator parameters
  - Using request map editor
  - Opening/saving actuation map file
  - General actuator parameters

# WHAT IS GAA?

- **A General Actuation Analysis (GAA) typically applies forces/motions directly to the spindles of a full vehicle assembly. These inputs are called actuators.**
- **The actuator functions can be defined and saved for later re-use.**
- **RPC generation and content definition can be performed via request map files.**
- **Earlier this was done manually using command line to setup actuators in the standard interface and Adams/View mode for simulation control.**
- **This method is integrated into and maintained in Adams/Car and saves lot of time compared to the manual approach.**

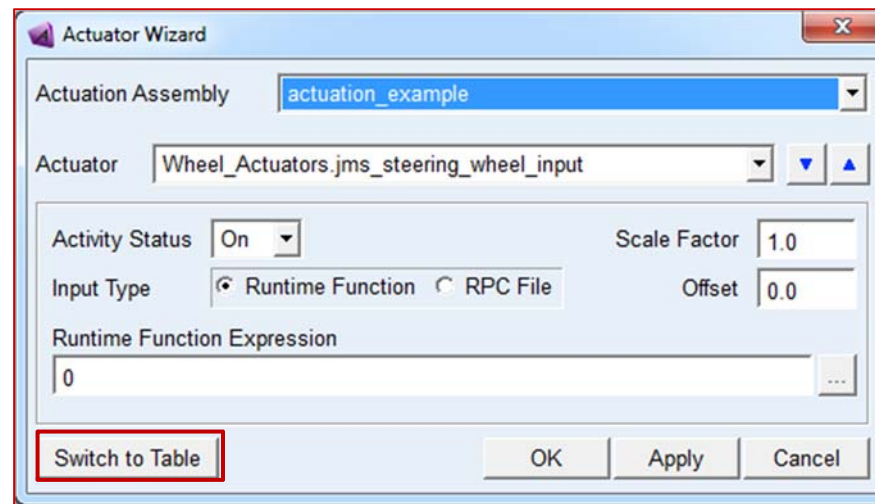


# GAA MODEL DETAILS

- **A modified version of the conventional full-vehicle assembly in Adams/Car is used in the General Actuation Analysis functionality. This assembly defined is of the ASSEMBLY\_CLASS 'Actuation'.**
- **The database needs to be modified to include the folder 'actuation\_inputs.tbl'. The wheel template for this assembly also needs to be defined differently without reference to a tire property file. The actuation inputs are applied directly to the wheel part without tire contact.**

# ACCESSING ACTUATOR PARAMETERS

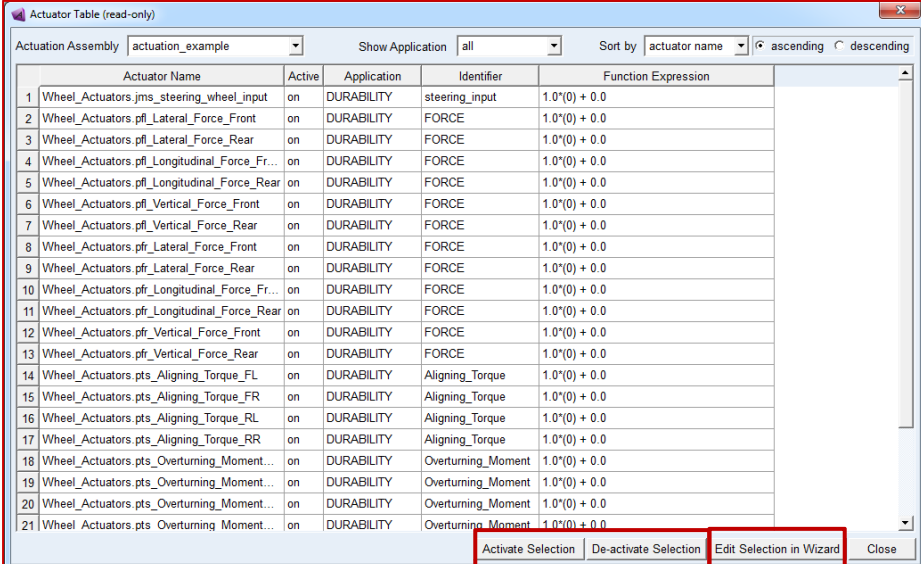
- You are provided with the option of using a wizard of a tabular interface for editing actuator data. These options can be accessed through the Adjust → Actuators → Table/Wizard menus. The wizard interface is shown below:



- The desired actuator can be selected either through the drop down or using the up-down arrow buttons. Parameter data related to the selected actuator is displayed in the dialog and can be modified.

# ACCESSING ACTUATOR PARAMETERS

- The Switch to Table button is provided to switch to a table view from the wizard.
- The table-like interface provides an easy way of looking at a list of actuators at once and activating or deactivating an entire selection.
- The desired set of actuators is selected first. Using the 'Activate Selection' or 'De-activate Selection' buttons, the selected set can be activated or de-activated at once.



Actuator Table (read-only)

Actuation Assembly: actuation\_example Show Application: all Sort by: actuator name ascending

	Actuator Name	Active	Application	Identifier	Function Expression
1	Wheel_Actuators.jms_steering_wheel_input	on	DURABILITY	steering_input	1.0*(0) + 0.0
2	Wheel_Actuators.pfl_Lateral_Force_Front	on	DURABILITY	FORCE	1.0*(0) + 0.0
3	Wheel_Actuators.pfl_Lateral_Force_Rear	on	DURABILITY	FORCE	1.0*(0) + 0.0
4	Wheel_Actuators.pfl_Longitudinal_Force_Fr...	on	DURABILITY	FORCE	1.0*(0) + 0.0
5	Wheel_Actuators.pfl_Longitudinal_Force_Rear	on	DURABILITY	FORCE	1.0*(0) + 0.0
6	Wheel_Actuators.pfl_Vertical_Force_Front	on	DURABILITY	FORCE	1.0*(0) + 0.0
7	Wheel_Actuators.pfl_Vertical_Force_Rear	on	DURABILITY	FORCE	1.0*(0) + 0.0
8	Wheel_Actuators.pfr_Lateral_Force_Front	on	DURABILITY	FORCE	1.0*(0) + 0.0
9	Wheel_Actuators.pfr_Lateral_Force_Rear	on	DURABILITY	FORCE	1.0*(0) + 0.0
10	Wheel_Actuators.pfr_Longitudinal_Force_Fr...	on	DURABILITY	FORCE	1.0*(0) + 0.0
11	Wheel_Actuators.pfr_Longitudinal_Force_Rear	on	DURABILITY	FORCE	1.0*(0) + 0.0
12	Wheel_Actuators.pfr_Vertical_Force_Front	on	DURABILITY	FORCE	1.0*(0) + 0.0
13	Wheel_Actuators.pfr_Vertical_Force_Rear	on	DURABILITY	FORCE	1.0*(0) + 0.0
14	Wheel_Actuators.pts_Aligning_Torque_FL	on	DURABILITY	Aligning_Torque	1.0*(0) + 0.0
15	Wheel_Actuators.pts_Aligning_Torque_FR	on	DURABILITY	Aligning_Torque	1.0*(0) + 0.0
16	Wheel_Actuators.pts_Aligning_Torque_RL	on	DURABILITY	Aligning_Torque	1.0*(0) + 0.0
17	Wheel_Actuators.pts_Aligning_Torque_RR	on	DURABILITY	Aligning_Torque	1.0*(0) + 0.0
18	Wheel_Actuators.pts_Overturning_Moment...	on	DURABILITY	Overturning_Moment	1.0*(0) + 0.0
19	Wheel_Actuators.pts_Overturning_Moment...	on	DURABILITY	Overturning_Moment	1.0*(0) + 0.0
20	Wheel_Actuators.pts_Overturning_Moment...	on	DURABILITY	Overturning_Moment	1.0*(0) + 0.0
21	Wheel_Actuators.pts_Overturning_Moment...	on	DURABILITY	Overturning_Moment	1.0*(0) + 0.0

Activate Selection De-activate Selection Edit Selection in Wizard Close

# USING REQUEST MAP EDITOR

- **The request map editor**
  - View or edit requests that are defined in the database
  - Import a request map file and edit the requests defined there.
- **Tool → Requests → Request Map Editor**

The screenshot shows the 'Request Map Editor' window. At the top, there are dropdown menus for 'Assembly' (set to 'actuation\_example') and 'Request Map' (set to 'req2rpc'). Below these is a text field for 'RPC Filename Prefix'. A section labeled 'Populate Table (\* appends)' contains a red box around the 'All' button, another red box around the '\* Browse ...' button, and an 'Import:' button with a file icon. The main area is a table with the following data:

	Request Name	Components	Major Role	Minor Role
82	toe_camber	2468	suspension	rear
83	bgr_subframe_front.disp_request	12345678	suspension	rear
84	bgr_subframe_front.velo_request	12345678	suspension	rear
85	bgr_subframe_front.force_requ...	12345678	suspension	rear
86	bgl_subframe_rear.disp_request	12345678	suspension	rear
87	bgl_subframe_rear.velo_request	12345678	suspension	rear
88	bgl_subframe_rear.force_request	12345678	suspension	rear
89	bgr_subframe_rear.disp_request	12345678	suspension	rear
90	bgr_subframe_rear.velo_request	12345678	suspension	rear

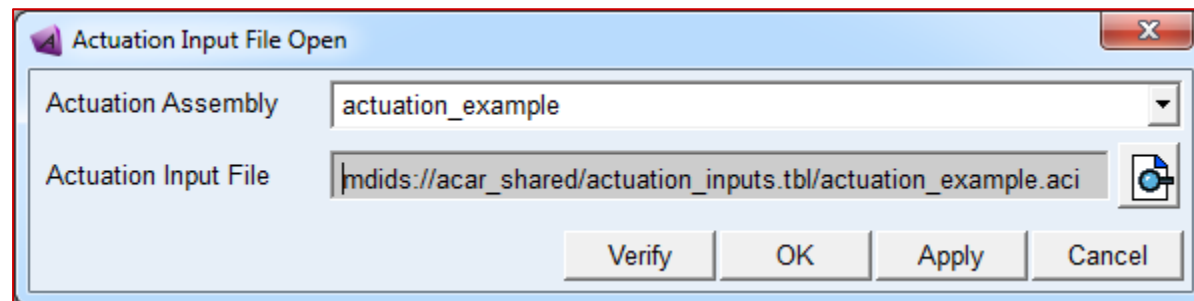
Below the table, there are controls for 'Clear table', 'Sort by' (set to 'request name'), and radio buttons for 'ascending' (selected) and 'descending'. A red box highlights the 'Edit Mode' checkbox, which is currently unchecked. Below this is a 'Filter by' section with radio buttons for 'Major Role' (selected) and 'Minor Role', and a dropdown set to 'all'. At the bottom, there is a 'Target' dropdown set to 'Database', a text field for 'acar\_training', and an 'Info' button. A 'New request map name' text field is also present. At the very bottom, there are three buttons: 'Apply to Model' (highlighted with a red box), 'Save to Request Map File' (highlighted with a red box), and 'Cancel'.

# USING REQUEST MAP EDITOR


- **In the Database mode, you can select to display**
  - all requests by clicking **All**
  - browse for a set of requests from the assembly by clicking **Browse**.
- **In order to edit the request data in the table, you must explicitly switch to the edit mode by checking the Edit box.**
- **You may choose to apply the changes to**
  - the model directly
  - a request map file.

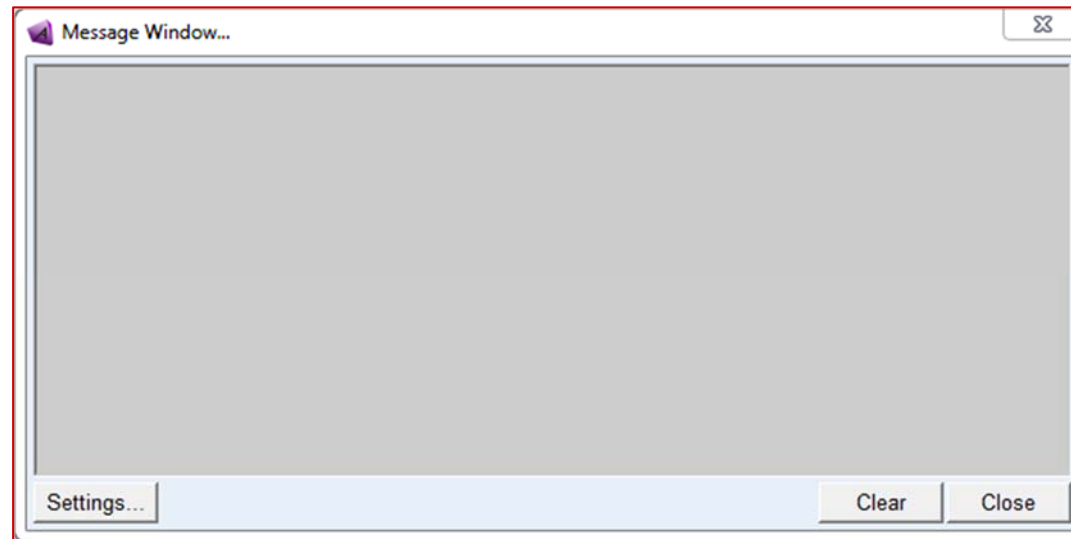
# OPENING/SAVING ACTUATION INPUT MAP FILE

- An actuation input map file contains a series of parameter-value pairs corresponding to one or more actuators in an assembly.
- The parameters include major and minor roles of the actuator, scale factor, offset, time offset, RPC function, etc. Such a file provides an easy way of setting up a large number of actuators at once.
- A facility has been provided to let you view/open an actuation input map file or create one using actuator data from a valid assembly.
- The open/save functionality can be accessed from the Simulate menu, select General Actuation Analysis → Actuation Input File.



# OPENING/SAVING ACTUATION INPUT MAP FILE

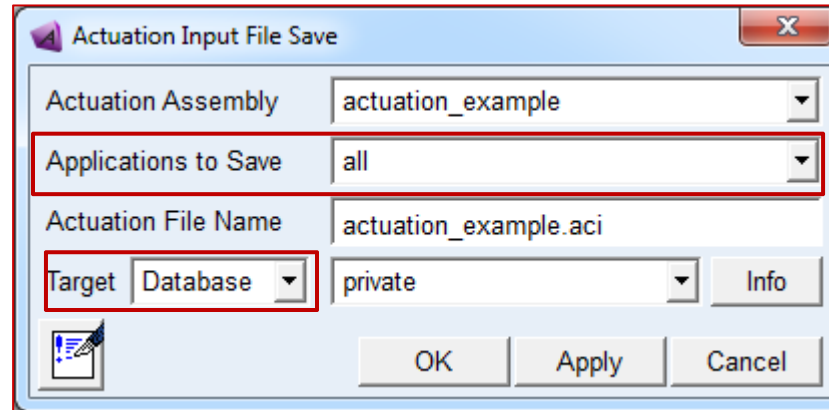
- **NOTE:** It's not necessary to have any valid active assemblies open in the current session in order to view an existing actuation input file.
- Select the desired file and click the button with magnifying glass  to open the file in message window.
- **The verify functionality reports**
  - the actuators present in the file that are absent in the assembly
  - incorrect roles set for the actuator, if any.



# OPENING/SAVING ACTUATION INPUT MAP FILE

- The actuator settings changed manually for desired actuators can be saved into a database:

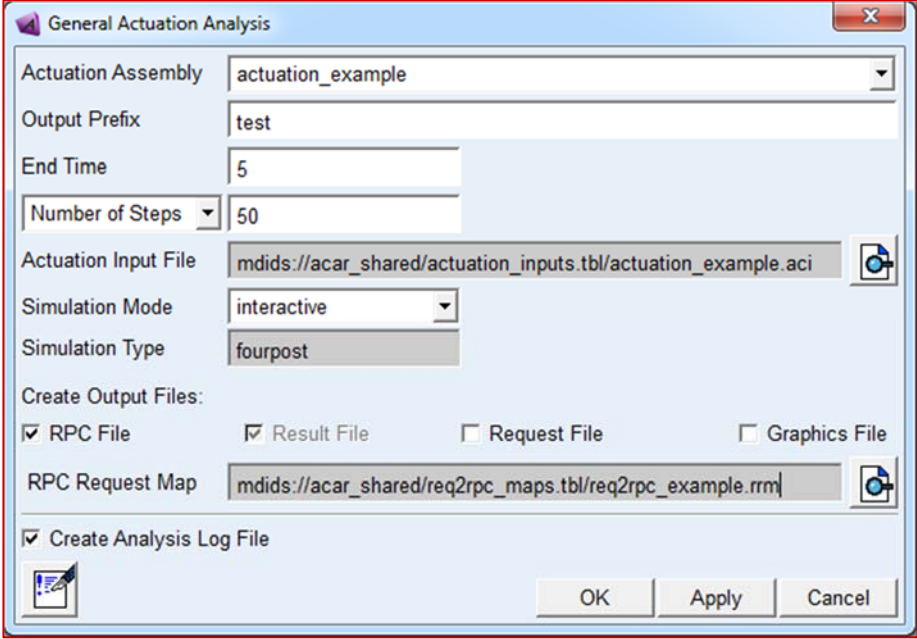
**General Actuation Analysis → Actuation Input File → Save**





# GENERAL ACTUATION PARAMETERS

Output Control  
Parameters



The screenshot shows the 'General Actuation Analysis' dialog box with the following settings:

- Actuation Assembly: actuation\_example
- Output Prefix: test
- End Time: 5
- Number of Steps: 50
- Actuation Input File: mdirs://acar\_shared/actuation\_inputs.tbl/actuation\_example.aci
- Simulation Mode: interactive
- Simulation Type: fourpost
- Create Output Files:
  - ☒ RPC File
  - ☒ Result File
  - ☐ Request File
  - ☐ Graphics File
- RPC Request Map: mdirs://acar\_shared/req2rpc\_maps.tbl/req2rpc\_example.rmq
- ☒ Create Analysis Log File

Buttons at the bottom: OK, Apply, Cancel.

Simulation  
Parameters

Actuator  
Set-up

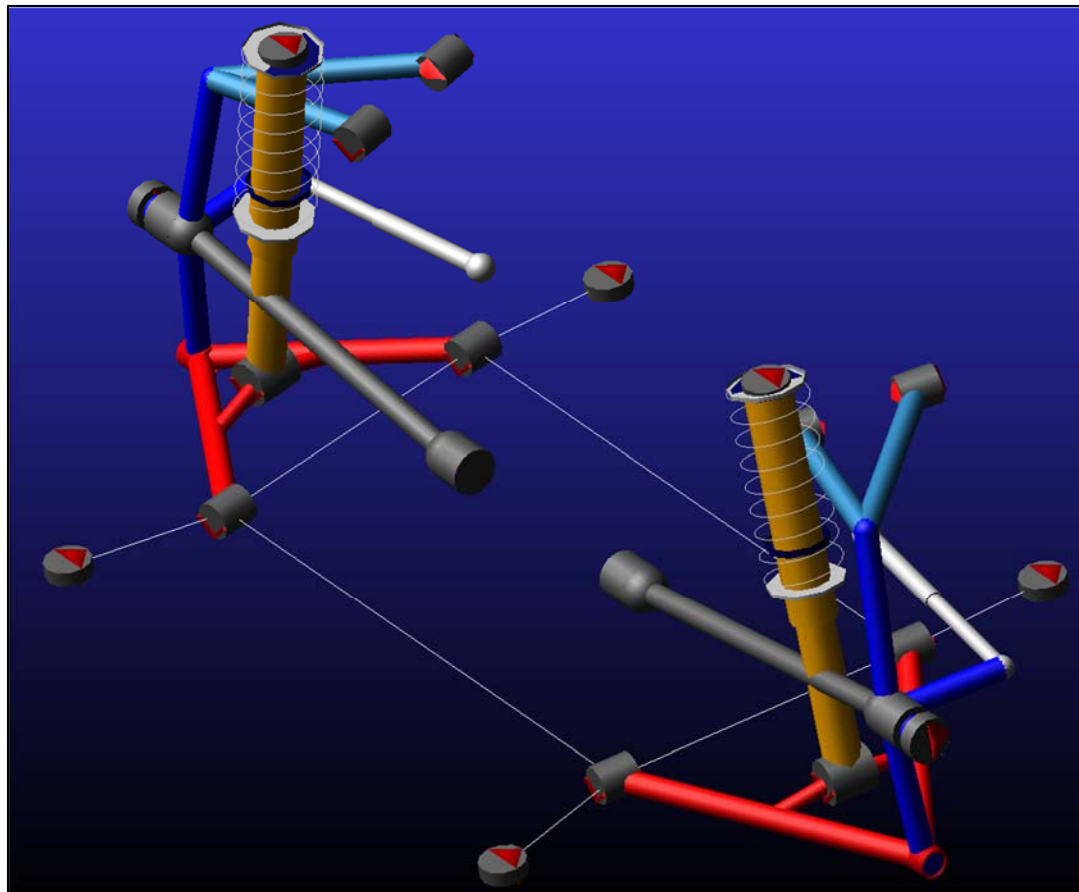
- **The input parameters can be categorized into 3 classes:**
  - Simulation Parameters :To define the simulation end time, mode etc.
  - Actuator parameters :To Set-up actuator parameters.
  - Output Control Parameters :To define the output files generated as part of the analysis.

# EXERCISE

- **Perform Workshop 11, “GENERAL ACTUATION ANALYSIS”.**

## SECTION 12

# PARAMETERIZATION & BUILDING TEMPLATES



# BUILDING TEMPLATES

- **What's in this section:**
  - Parameterization
    - Parameterization in Adams/Car
    - Creating Hardpoints
    - Creating Construction Frames
    - Location Parameterization
    - Orientation Parameterization
  - Building Templates
    - Template Overview
    - Template Topology
    - File Architecture
    - Building a New Template
    - Types of Parts
    - Rigid and Flexible Bodies (Parts)

# BUILDING TEMPLATES

- **What's in this section (Cont.):**
  - Building Templates (cont.)
    - Geometry
    - Attachments (Joints and Bushings)
    - Springs
    - Dampers
    - Bumpstops and Reboundstops
    - Toe/Camber Angles and Suspension Parameter Array
    - Adjustable Forces
    - Parameter Variables
    - General Advice



# PARAMETERIZATION

# PARAMETERIZATION IN ADAMS/CAR

- **Why parameterize?**

- Parameterizing a template allows you
  - to build relationships into the model so that when you change a modeling entity, Adams/Car automatically updates all other entities that depend on it.
  - to build a whole vehicle model to depend on only a few key hardpoints and variables, saving time and effort in making design changes.

- **What can you parameterize?**

- Location and orientation expressions
- Geometry
- Group activity
- Functions
- And so on

# CREATING HARDPOINTS

- **Hardpoints allow you to:**
  - Define key locations in your model.
  - Parameterize locations for higher-level entities, such as construction frames, parts, and attachments.
  - **Note:** Hardpoints are the same as points in Adams/View
- **Creating hardpoints in Template Builder:**
  - Go to **Build → Hardpoint → New**. In the dialog box, specify the name of the hardpoint, if it's a left, right or single, and the location.
  - When you create a hardpoint of type left or right, Adams/Car creates a corresponding hardpoint reflected across the car's longitudinal axis.

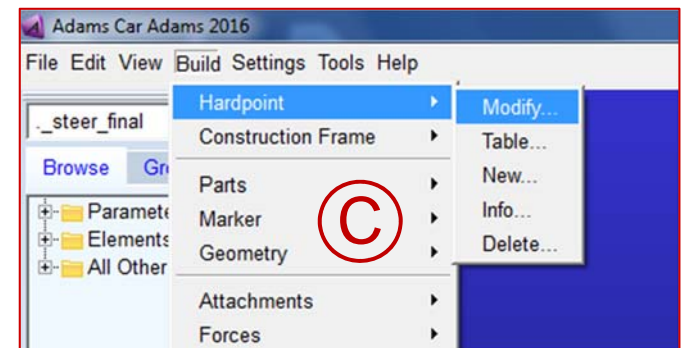
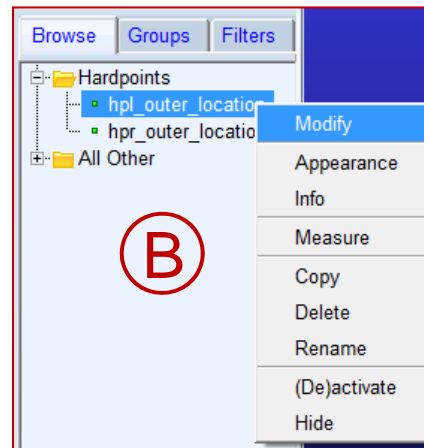
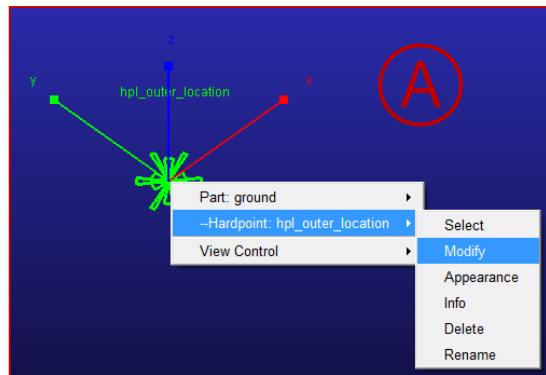


# MODIFYING HARDPOINTS

- **Different ways to modify a hardpoint:**

- A. Right-click the hardpoint and select the hardpoint name followed by Modify.

- B. From the Model Browser: point to **Hardpoint**, right click the hardpoint and then select **Modify**.



- C. From the menu: **Build** → **Hardpoint**, where you can select:

- **Table** - Every hardpoint in the model appears and you can enter new locations for all.

- **Modify** - You can modify only one hardpoint at a time.

# CREATING CONSTRUCTION FRAMES

- **Construction frames are:**

- Building blocks that you use whenever an entity requires that you specify an orientation in addition to a location.

**Note:** Construction frames are the same as markers in Adams/View

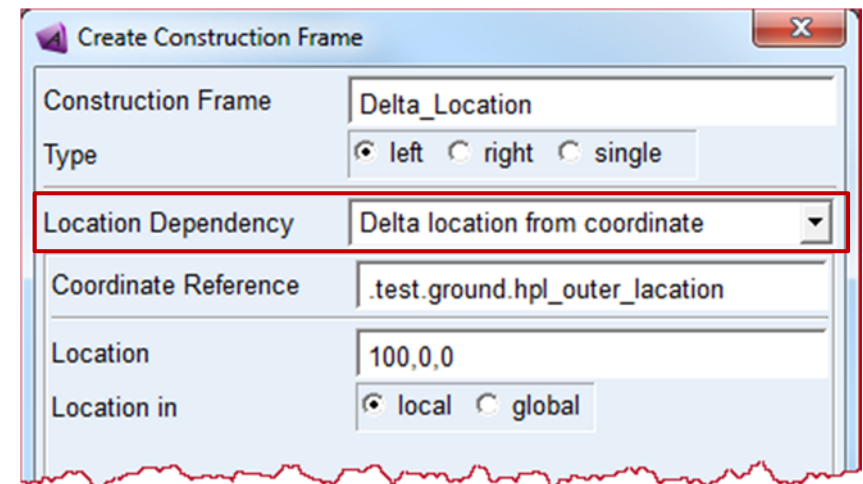
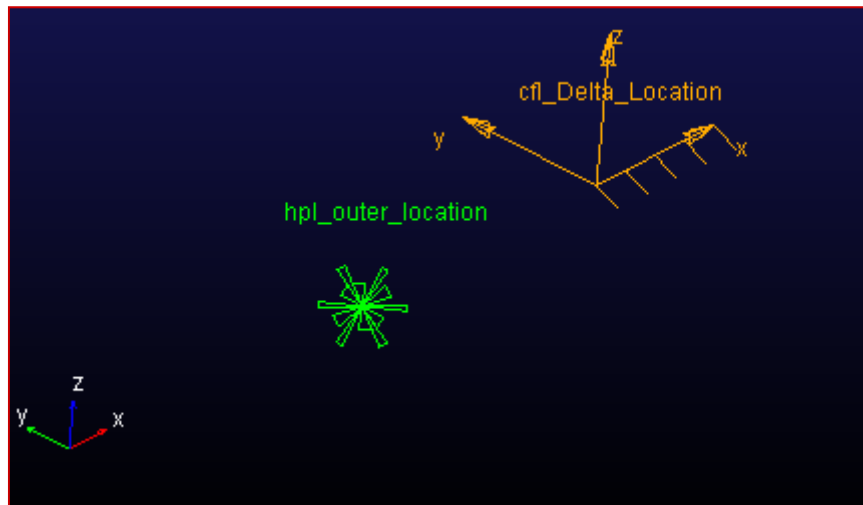
- **Creating construction frames in Template Builder:**

- Go to **Build → Construction Frame → New**. In the dialog box, specify the name and if it's a left, right, or single.
- Locations and orientations can be defined in many ways as shown in the following slides.

# LOCATION PARAMETERIZATION

- **Delta location from coordinate**
  - Locate with respect to a defined reference frame.
  - You can define X, Y, Z displacement in the local or global coordinate reference frame.

Location is 100 length units along global x-axis

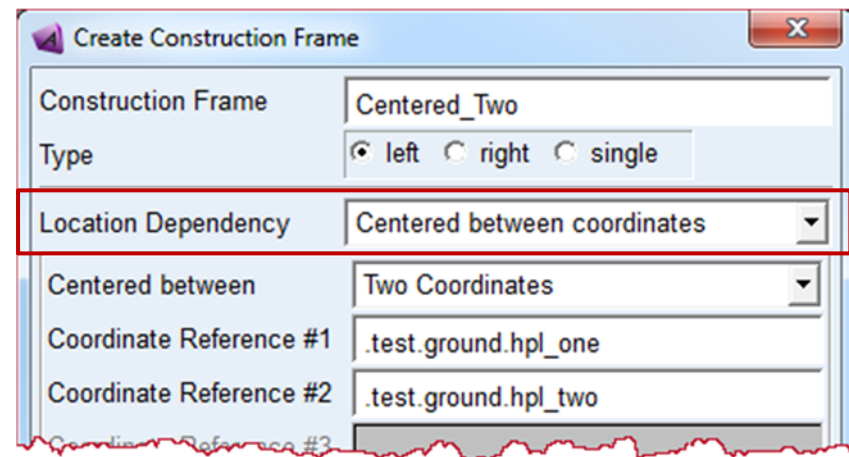
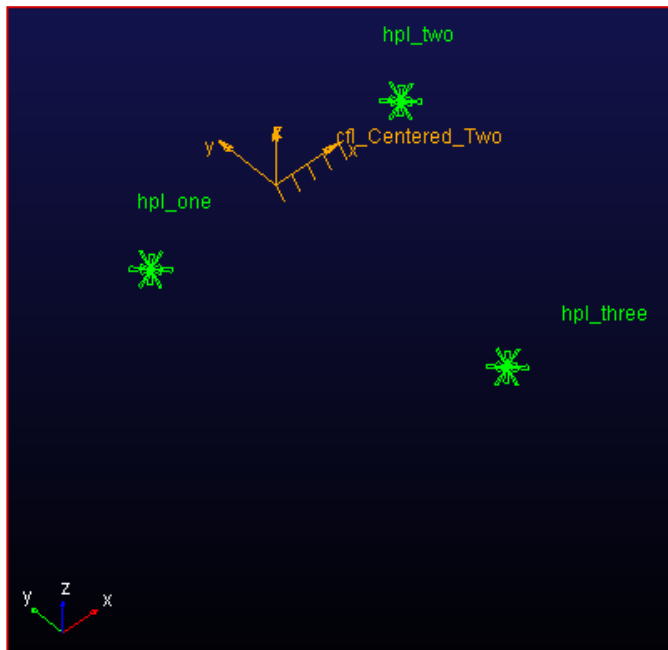


Global reference frame      Coordinate reference

# LOCATION PARAMETERIZATION

- **Centered between coordinates**

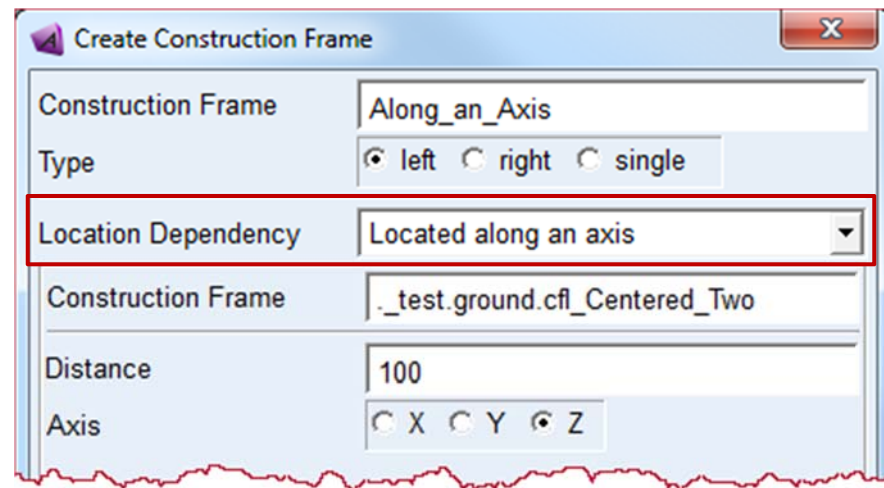
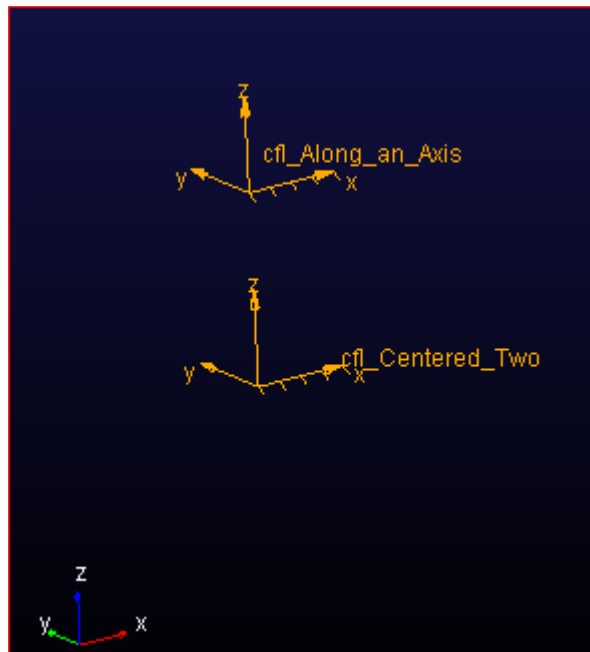
- Using the two-coordinates method, the entity is located on the mid-point along an imaginary line joining the defined reference coordinates.
- Using the three-coordinate method, the entity is located on the center point of a plane defined by the three reference coordinates.



# LOCATION PARAMETERIZATION

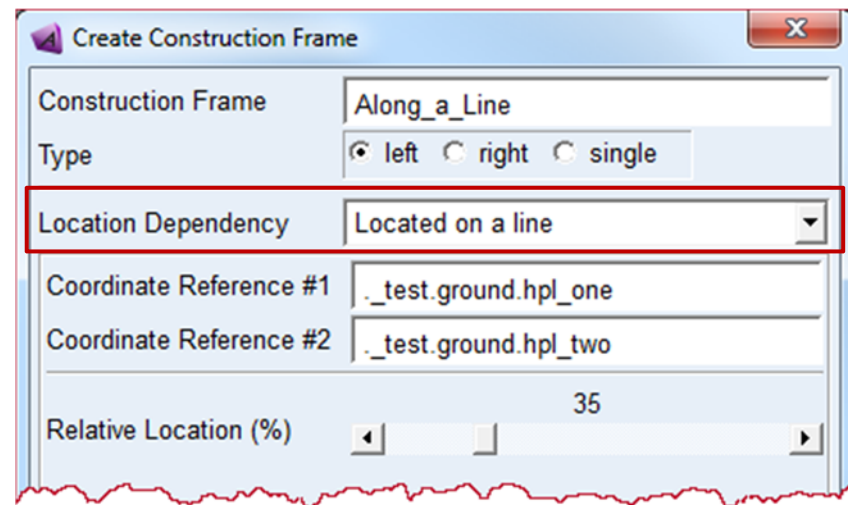
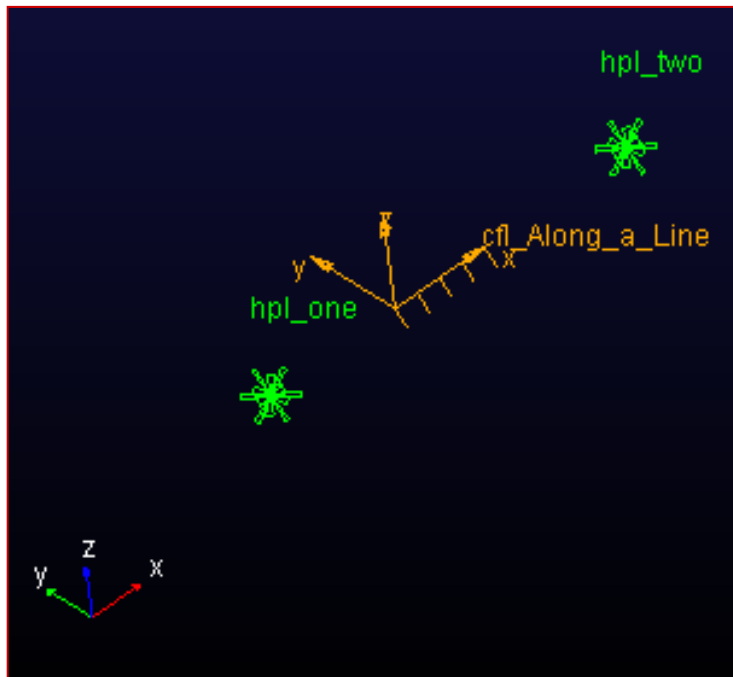
- **Located along an axis**

- The entity is located a defined distance along the chosen axis of the reference construction frame.
- In this example, the entity has been located 100 length units along the z-axis of the reference construction frame.



# LOCATION PARAMETERIZATION

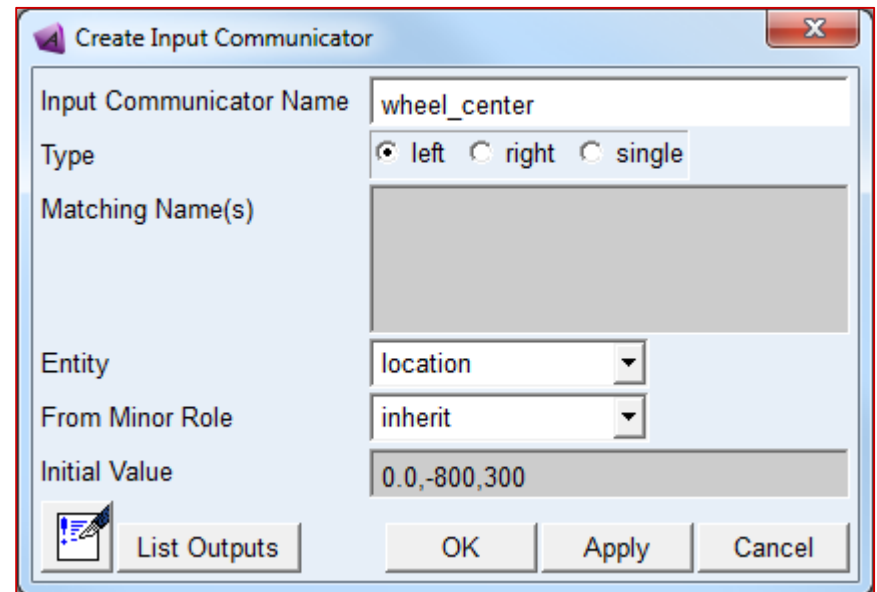
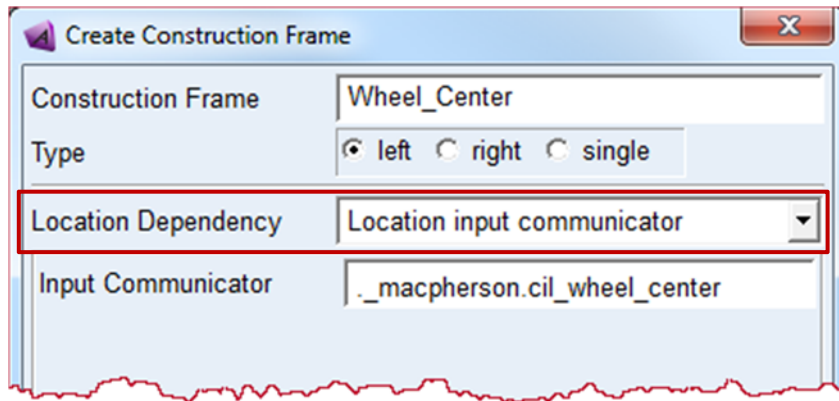
- **Located on a line**
  - Locates the entity along a line defined by the two reference coordinates.
  - The entity is located a defined percentage along the line measured from the first reference coordinate to the second reference coordinate.



# LOCATION PARAMETERIZATION

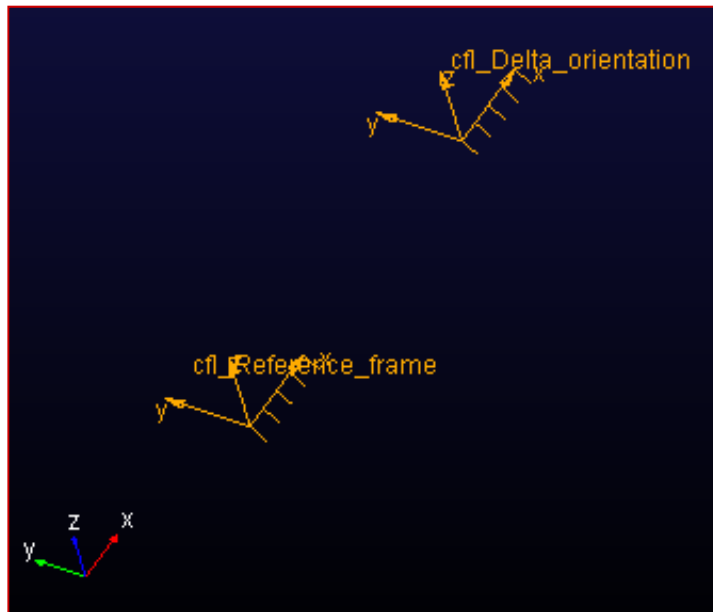
- **Location input communicator**

- Locates the entity using location data from the chosen input communicator.
- Use this option to locate entities with respect to reference frames in other templates.
- At the assembly stage, the location data is communicated to the input communicator from the corresponding output communicator in the other template. Up to that point, the entity is located at the initial value defined in the input communicator.



# ORIENTATION PARAMETERIZATION

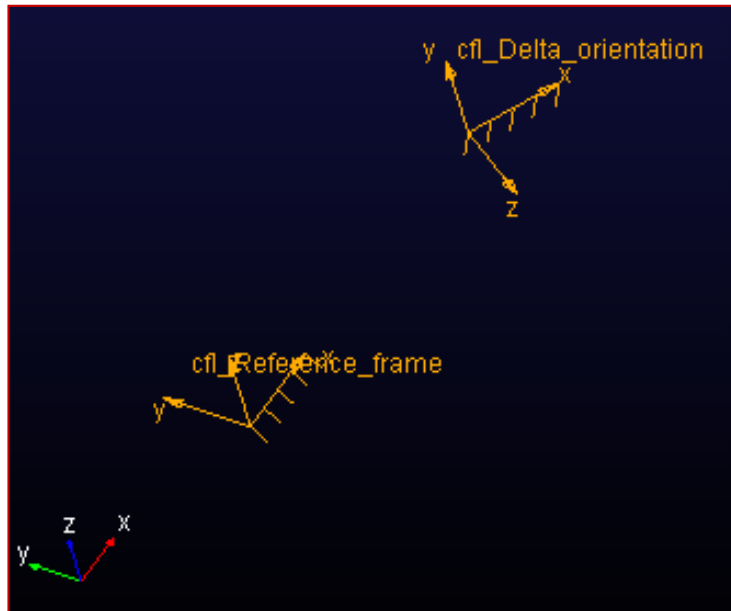
- **Delta orientation from coordinate**
  - Orients the entity with respect to the reference frame, using the Euler angle orientation defined.
  - In this example, the entity has been defined such that there is no change in orientation with respect to the reference construction frame.





# ORIENTATION PARAMETERIZATION

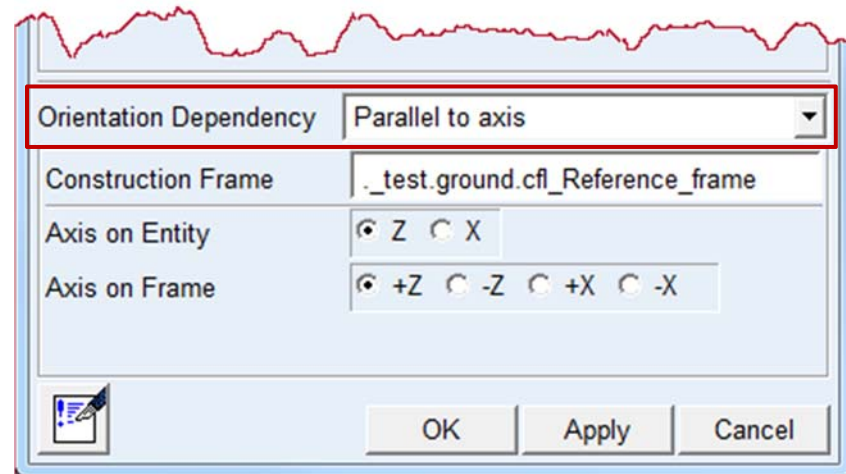
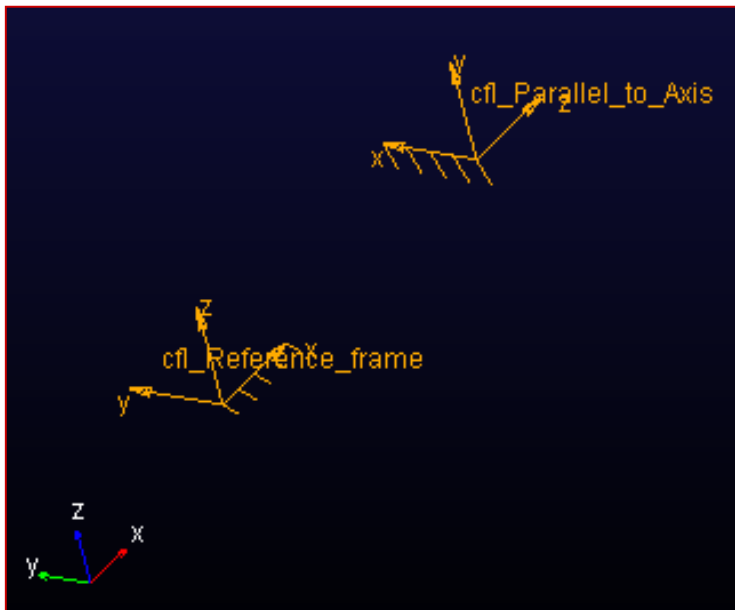
- **Delta orientation from coordinate**
  - Orients the entity with respect to the reference frame, using the Euler angle orientation defined.
  - In this example, the entity has been rotated -30 degrees about the reference frame z-axis and then 90 degrees about the new x-axis.



# ORIENTATION PARAMETERIZATION

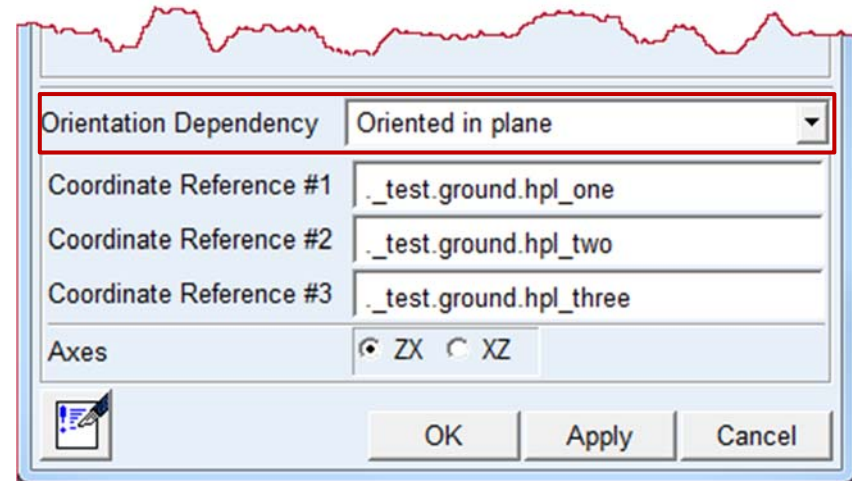
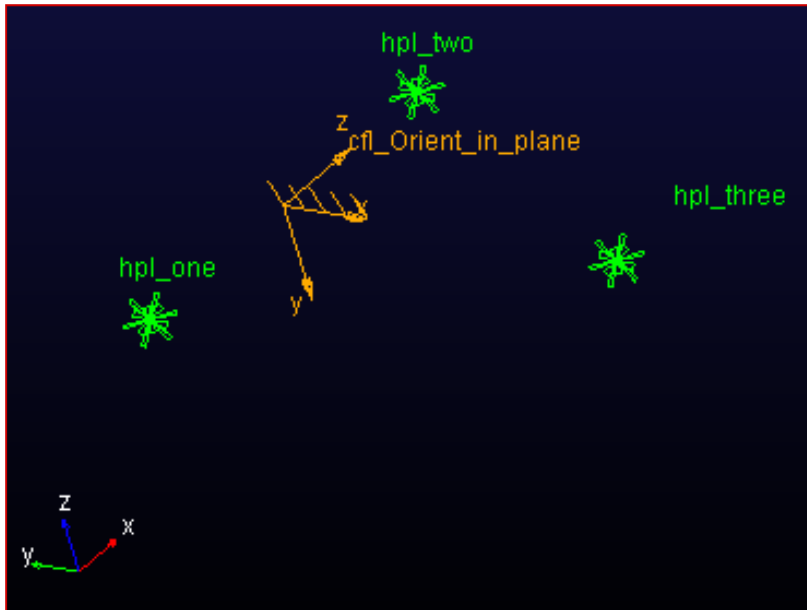
- **Parallel to axis**

- Orients the defined axis on the entity parallel to the defined axis on the reference frame.
- In this example, the z-axis of the entity is oriented parallel to the positive x-axis on the reference frame.



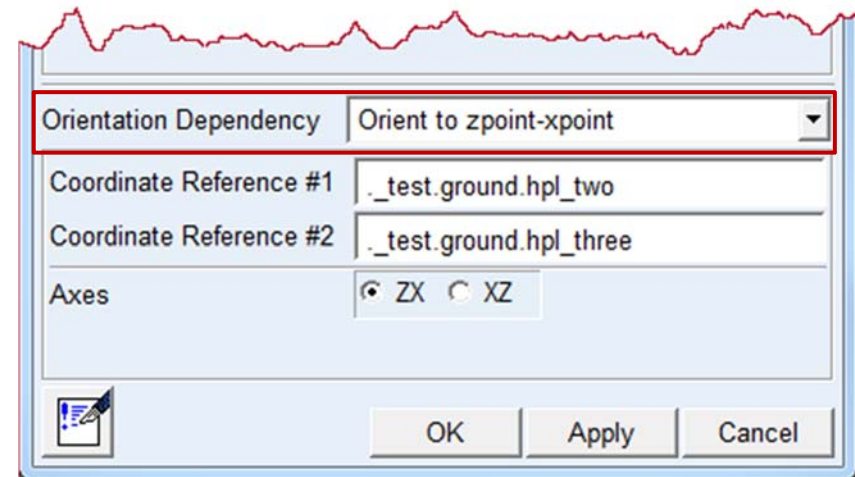
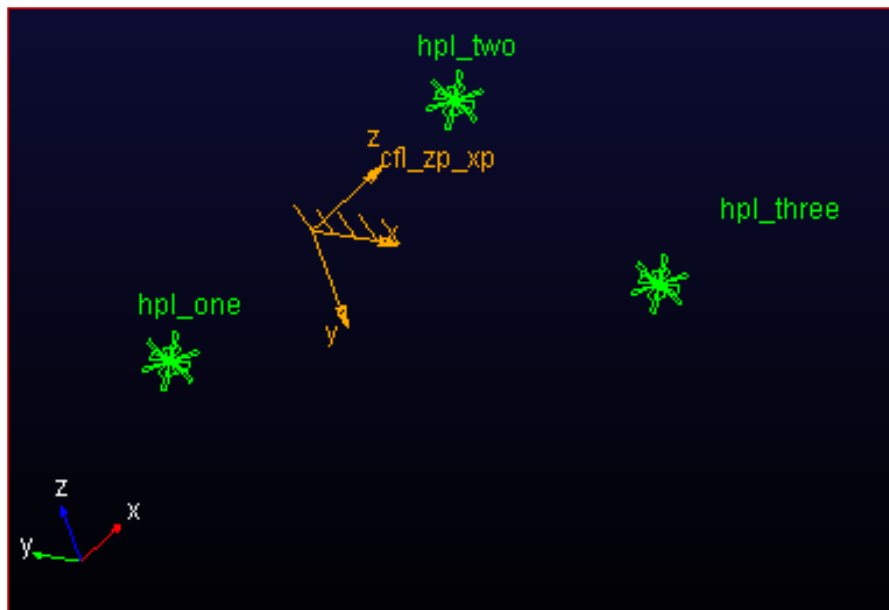
# ORIENTATION PARAMETERIZATION

- **Oriented in plane**
  - Orients the z-axis along a line defined by the first and second reference coordinates, and orients the x-axis such that the entity's zx plane lies in the plane defined by the three reference coordinates.



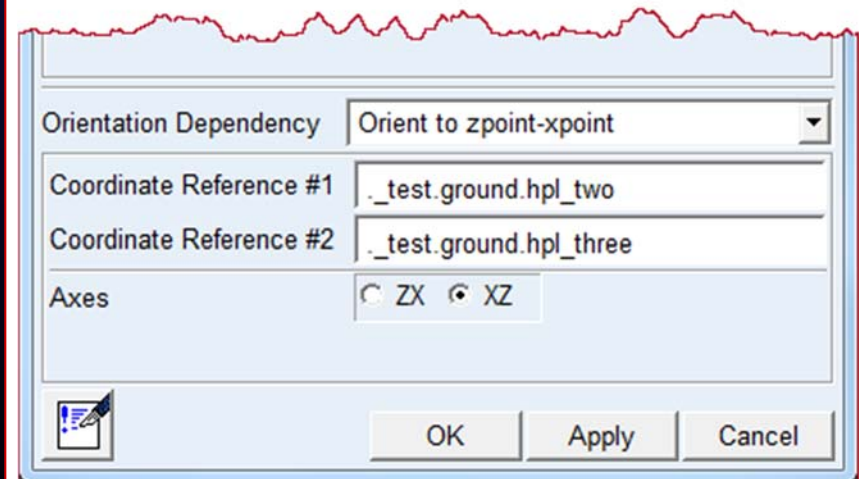
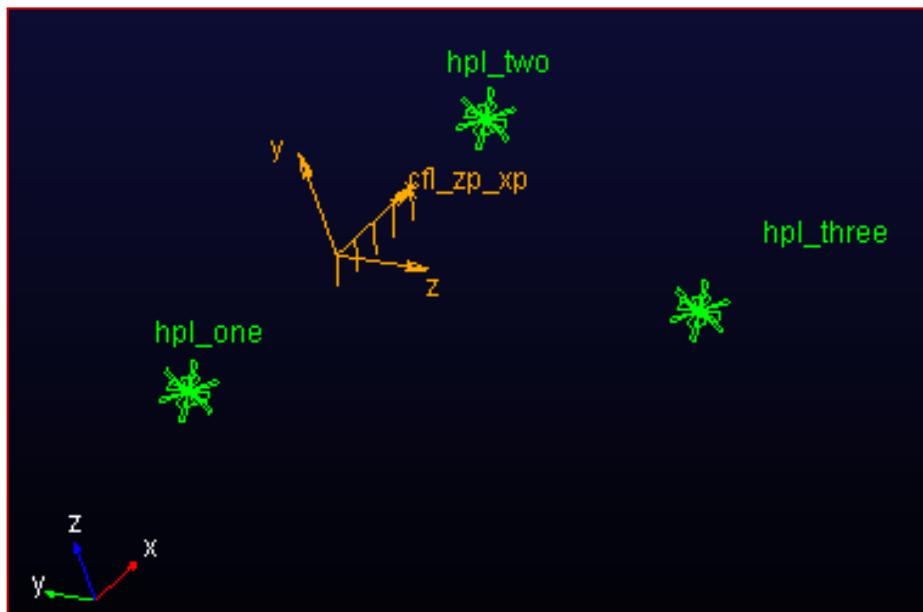
# ORIENTATION PARAMETERIZATION

- **Orient to zpoint-xpoint**
  - Orients the z-axis towards the first reference coordinate and the x-axis towards the second reference coordinate.



# ORIENTATION PARAMETERIZATION

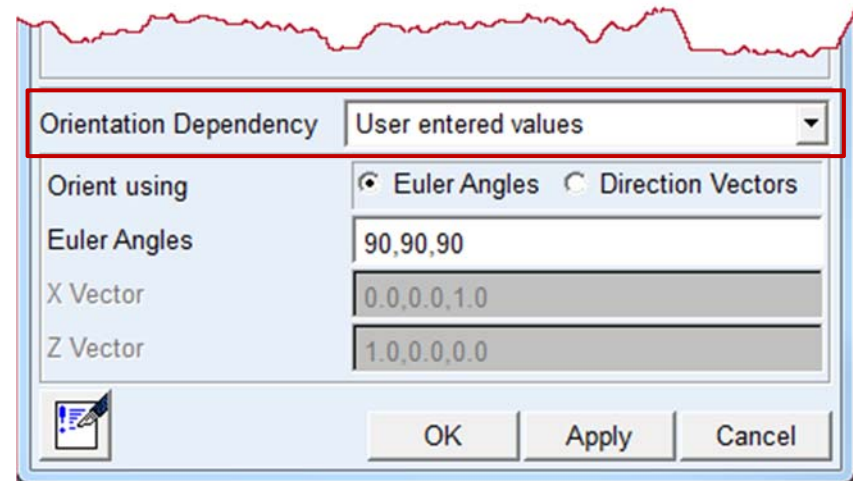
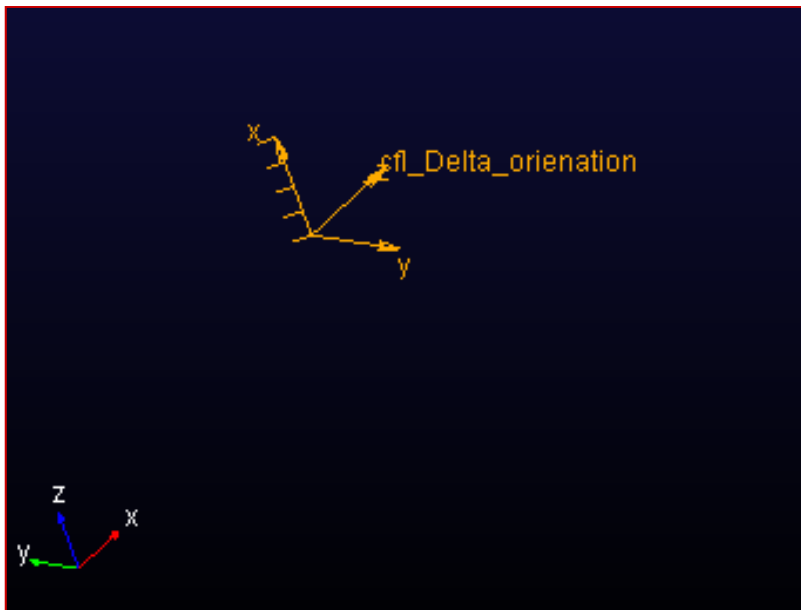
- **Orient to xpoint-zpoint**
  - Orients the x-axis towards the first reference coordinate and the z-axis towards the second reference coordinate.



# ORIENTATION PARAMETERIZATION

- **User-entered values**

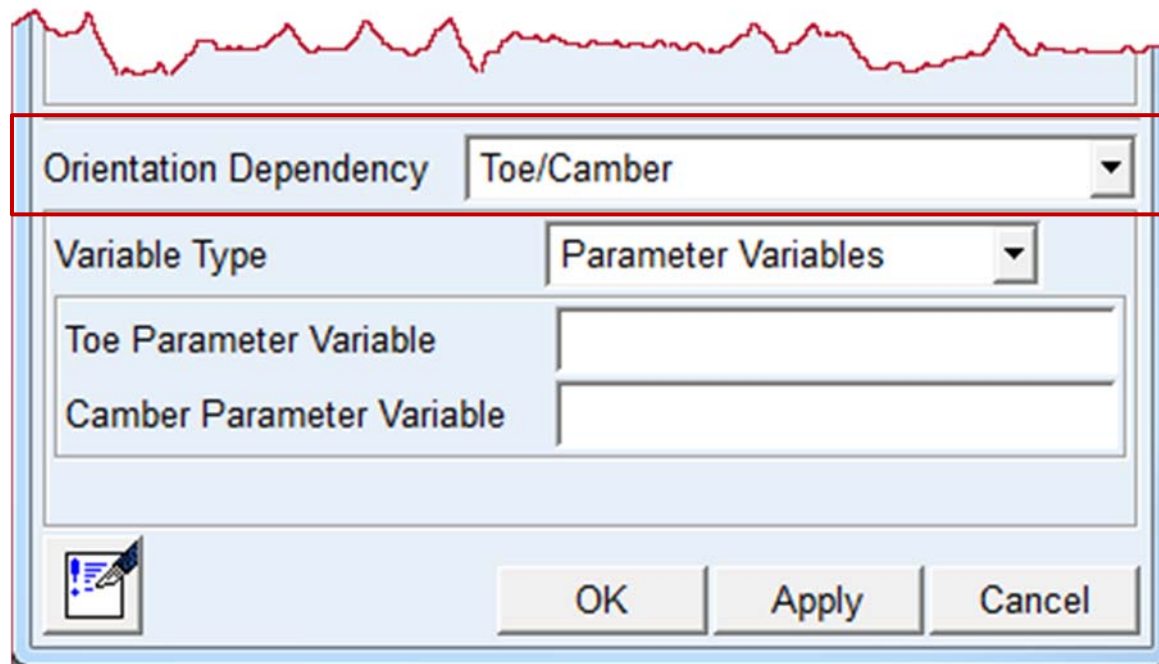
- Orients the construction frame using Euler angles: Z-X'-Z' incremental rotations with respect to the global construction frame (origo).



# ORIENTATION PARAMETERIZATION

- **Toe/Camber**

- Sets the toe and camber orientation. Generally, the toe and camber define the global orientation of the wheel spin axis. You obtain toe and camber values from parameter variables or input communicators.

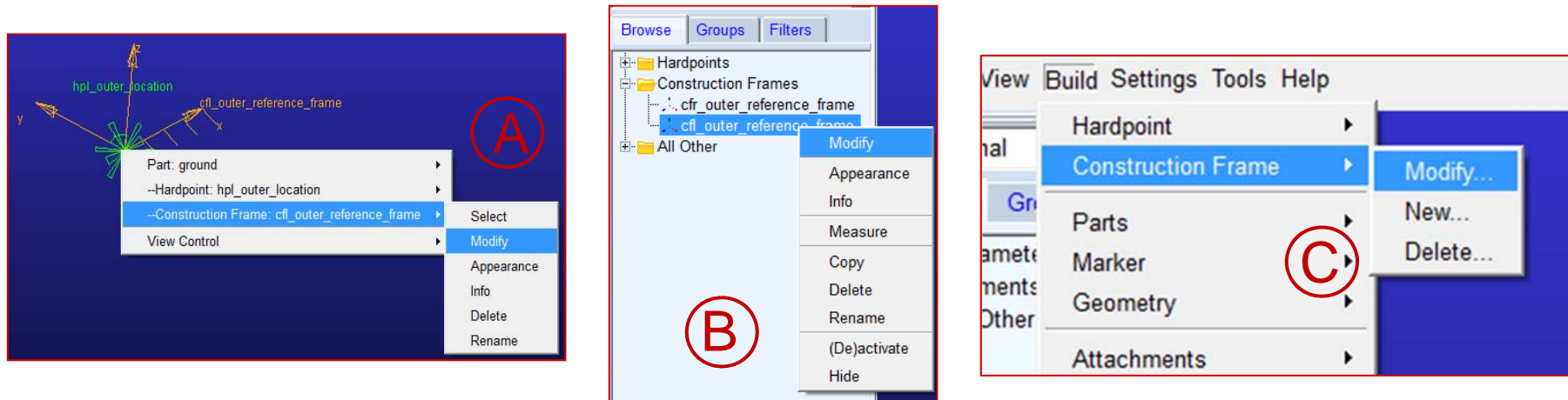


The image shows a software dialog box for configuring orientation parameters. At the top, there is a red jagged line representing a terrain profile. Below this, the dialog is divided into sections. The first section, labeled 'Orientation Dependency', contains a dropdown menu currently set to 'Toe/Camber'. The second section, labeled 'Variable Type', contains a dropdown menu set to 'Parameter Variables'. Below these are two input fields: 'Toe Parameter Variable' and 'Camber Parameter Variable', both of which are currently empty. At the bottom left of the dialog is a small icon of a document with a pencil. At the bottom right are three buttons: 'OK', 'Apply', and 'Cancel'.

# MODIFYING CONSTRUCTION FRAMES

- **Different ways to modify construction frames:**

- A. Right-click the construction frame and select **Modify**.
- B. From the Model Browser, point to **Construction Frames**, right click the **construction frame** and select **Modify**.



- C. From the menus, go to **Build → Construction Frame → Modify**. In the dialog box, select the name of the construction frame you want to modify.





# BUILDING TEMPLATES

# TEMPLATE OVERVIEW

- **Adams/Car templates are parameterized models in which you define:**
  - The topology of vehicle components, including parts, joints and force elements.
  - How the template communicates information to other templates and the test rig.
  - Note: A template could represent a single set of components or a complex collection of components.
- **At the template level:**
  - It is important to correctly define part connectivity and exchange of information because you cannot modify these at the subsystem level.
  - It is not important to correctly define the mass properties or force characteristics because these can be set at the subsystem level.
- **When building templates:**
  - Keep in mind the assembly process.
  - Make sure that your templates can communicate with each other and can communicate with the test rigs that you specify.

**Note:** Communicators define how the different subsystems exchange information.

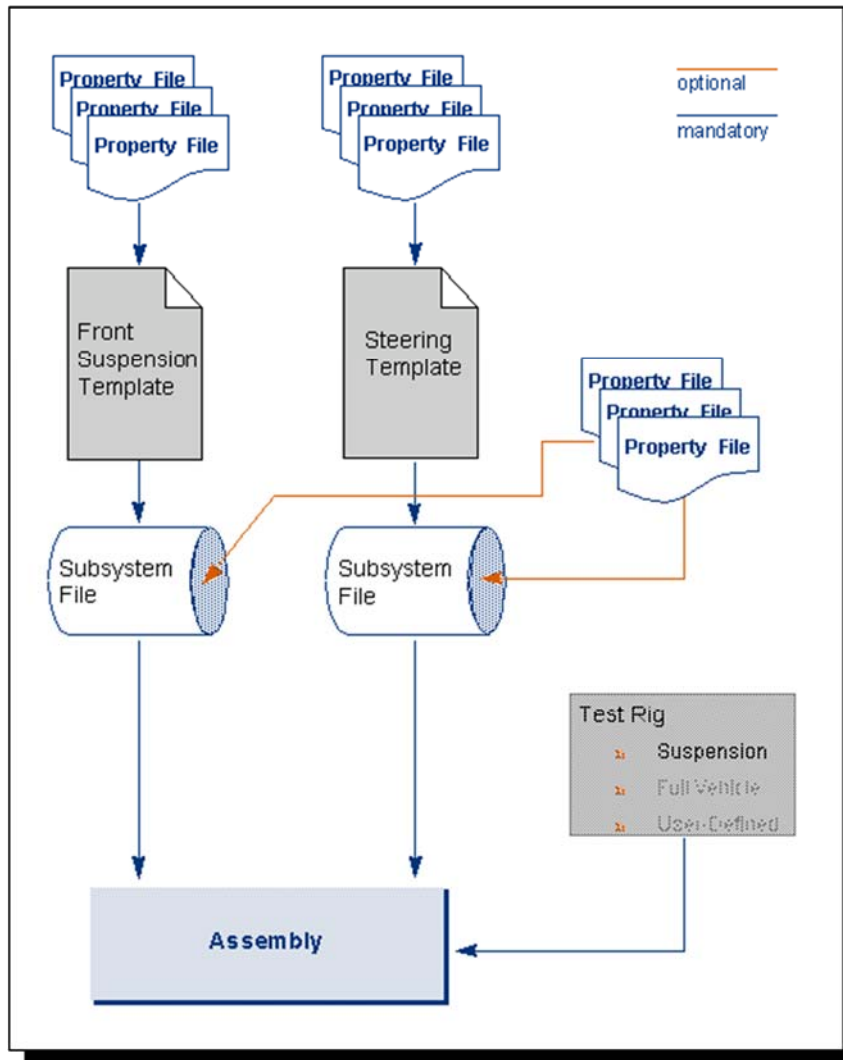
# DEFINING TEMPLATE TOPOLOGY

- **A model topology is defined via:**
  - Hardpoints and construction frames
    - You first create hardpoints and construction frames.
    - Hardpoints and construction frames are the Adams/Car elements that define all key locations and orientations in your model.
  - Parts
    - Once you've defined hardpoints and construction frames, you will use them to create parts.
  - Attachments
    - Create the attachments, such as joints and bushings, which tell Adams/Car how the parts react in relation to one another.
    - You can define attachments for the compliant and kinematic analysis modes.

# FILE ARCHITECTURE

- **The following files make up an Adams/Car database:**
  - Property files
  - Templates
  - Subsystems
  - Assemblies
- **The figure on the following page illustrates how template-based products use this architecture:**
  - Templates use property files to provide data for components such as springs, dampers, and bushings.
  - Subsystems, by default, reference property files found in the templates. Different property files can be specified at the subsystem level to override template default.
  - The Assembly is a collection of subsystems.

# FILE ARCHITECTURE



wheel\_envelopes.tbl  
 vehicle\_setups.tbl  
 torque\_converters.tbl  
 tires.tbl  
 templates.tbl  
 subsystems.tbl  
 steering\_assists.tbl  
 springs.tbl  
 shell\_graphics.tbl  
 roads.tbl  
 req2rpc\_maps.tbl  
 reboundstops.tbl  
 powertrains.tbl  
 plot\_configs.tbl  
 loadcases.tbl  
 leafsprings.tbl  
 gear\_stiffness.tbl  
 gear\_elements.tbl  
 flex\_bodys.tbl  
 driver\_roads.tbl  
 driver\_loadcases.tbl  
 driver\_knowledge.tbl  
 driver\_inputs.tbl  
 driver\_data.tbl  
 driver\_controls.tbl  
 differentials.tbl  
 dampers.tbl  
 bushings.tbl  
 bumpstops.tbl  
 assemblies.tbl  
 airsprings.tbl  
 aero\_forces.tbl  
 actuation\_inputs.tbl

Body.lt.sub  
 ISO\_Road.sub  
 MDI\_ANTI\_ROLL\_REAR.sub  
 MDI\_CHASSIS.sub  
 MDI\_FRONT\_STEERING.sub  
 MDI\_FRONT\_SUSPENSION.sub  
 MDI\_FRONT\_TIRES.sub  
 MDI\_REAR\_MULTI\_LINK.sub  
 MDI\_REAR\_SOLID\_AXLE\_SUSP.sub  
 MDI\_REAR\_SUSPENSION.sub  
 MDI\_REAR\_TIRES.sub  
 MDI\_REAR\_TWIST\_BEAM.sub  
 Powertrain.lt.sub  
 TR\_Body.sub

\_antiroll\_simple.tpl  
 \_brake\_system\_4Wdisk.tpl  
 \_double\_wishbone.tpl  
 \_double\_wishbone\_flex.tpl  
 \_double\_wishbone\_torsion.tpl  
 \_driveline\_fwd.tpl  
 \_driveline\_rwd.tpl  
 \_example\_leaf\_spring.tpl  
 \_handling\_tire.tpl  
 \_ISO\_road\_course.tpl  
 \_macpherson.tpl  
 \_mdi\_track.tpl  
 \_msc\_truck\_steer\_susp\_leaf.tpl  
 \_multi\_link.tpl

actuation\_example.asy  
 MDI\_Demo\_Vehicle.asy  
 MDI\_Demo\_Vehicle.lt.asy  
 mdi\_front\_vehicle.asy

# TEMPLATE COMPONENTS

- **To build a template, you typically use the following entities:**
  - Rigid bodies (parts)
  - Geometry
  - Attachments (joints and bushings)
  - Springs
  - Dampers
  - Bumpstops and reboundstops
  - The suspension parameter array
  - Communicators
- **Communicators are a very important part of Adams/Car that allow you to pass information from subsystem to subsystem, and are necessary for assemblies.**

# TYPES OF PARTS

- **Rigid bodies (parts)**
  - Movable parts with mass and inertia properties. Cannot deform.
- **Flexible bodies**
  - Movable parts with mass and inertia properties. Can bend when forces are applied to them.
- **Ground part**
  - Must exist in every model.
  - Defines the global coordinate system (GCS) and the global origin, and, therefore, remains stationary at all times.
  - Acts as the inertial reference frame for calculating velocities and acceleration.

# TYPES OF PARTS

- **Mount parts**
  - Massless parts that will be replaced by other parts in the assembly process.
- **Switch parts**
  - Massless parts and act like a switch for connections. By changing the switch part, one part will connect to another.
- **FE Part**
  - Movable parts with mass and inertia properties.
  - Can bend when forces are applied to them.
  - Composed of small pieces, known as finite elements.



# GEOMETRY

- **Geometry is used to enhance the visualization of a part using properties such as:**
  - Length
  - Radius
  - Width
  - Thickness
- **As opposed to parts, geometry is not necessary to perform simulations.**
- **You can have a part without geometry, but not geometry without a part.**

# ATTACHMENTS (JOINTS AND BUSHINGS)

- **Attachments join parts together and can define compliant or kinematic analysis modes.**
- **The compliant mode uses bushings, while the kinematic mode uses joints (for cases in which both are defined).**
  - **Bushings** - Provide three-dimensional forces and moments between parts, calculated with stiffness and damping. You can also specify preload and offset.
  - **Joints** - Provide a kinematic constraint between parts (infinitely rigid). You specify the type of joint that is applicable to your model.
- **You can place both a joint and a bushing as the same connection in a model. The joint will be used for kinematic analyses, and the bushing will be used for dynamic analyses: Adams/Car will automatically enable/disable based on the type of analysis.**

# SPRINGS

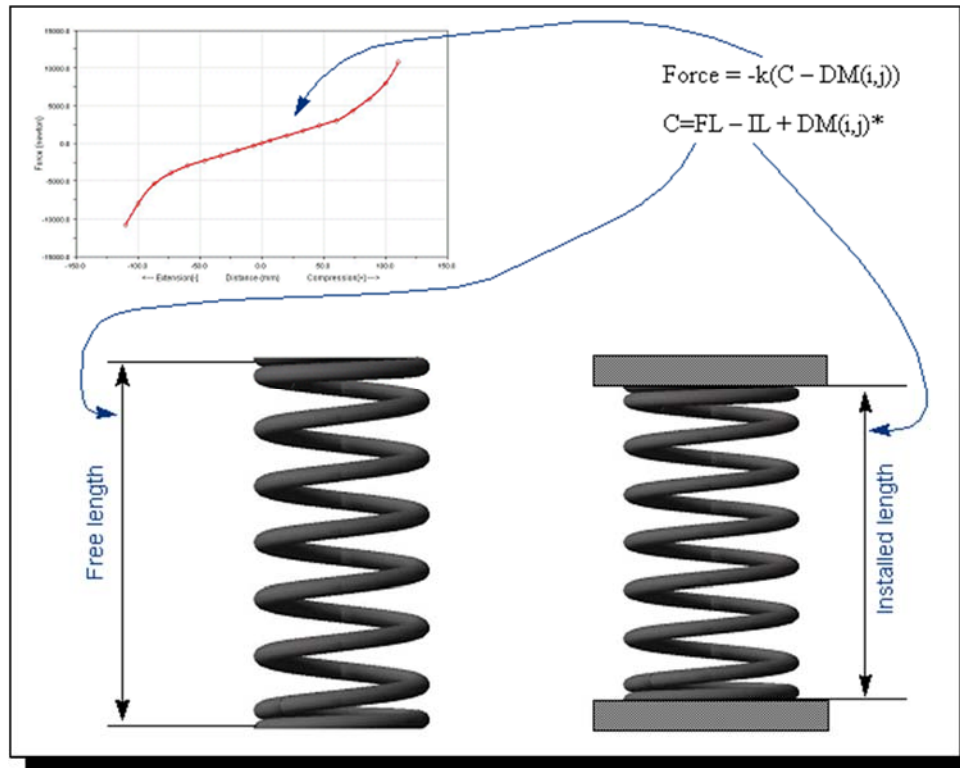
- When creating a spring, you need two coordinates to attach the two ends for the spring. The coordinates can be either hardpoints or construction frames. You must specify:
  - Two bodies between which you want the force to act and two reference frames (points at which the spring is attached on the bodies).
  - Installed length of the spring, which will be used to derive the design preload on the spring.
  - Property file, which contains the free length information, as well as the force/deflection characteristics.

The 'Create Spring' dialog box contains the following fields and controls:

- Spring Name: Text input field.
- I Part: Text input field.
- J Part: Text input field.
- I Coordinate Reference: Text input field.
- J Coordinate Reference: Text input field.
- Installed Length: A dropdown menu and a text field containing the formula  $DM(iCoord, jCoord)$ .
- Property File: Text input field containing `mdids://acar_shared/springs.tbl/mdi_000`.
- Color: Dropdown menu set to 'white'.
- Spring Diameter: Slider control set to 30.
- Number of Coils: Slider control set to 5.
- Bottom icons: Document, Graph, and Spring.
- Buttons: OK, Apply, and Cancel.

# SPRINGS

- Adams/Car calculates the force exerted by the spring using the equations shown in the figure.

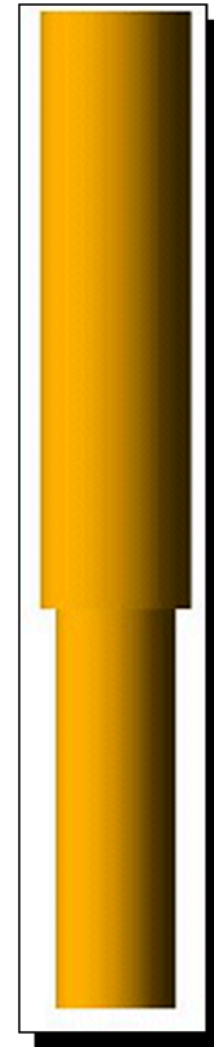
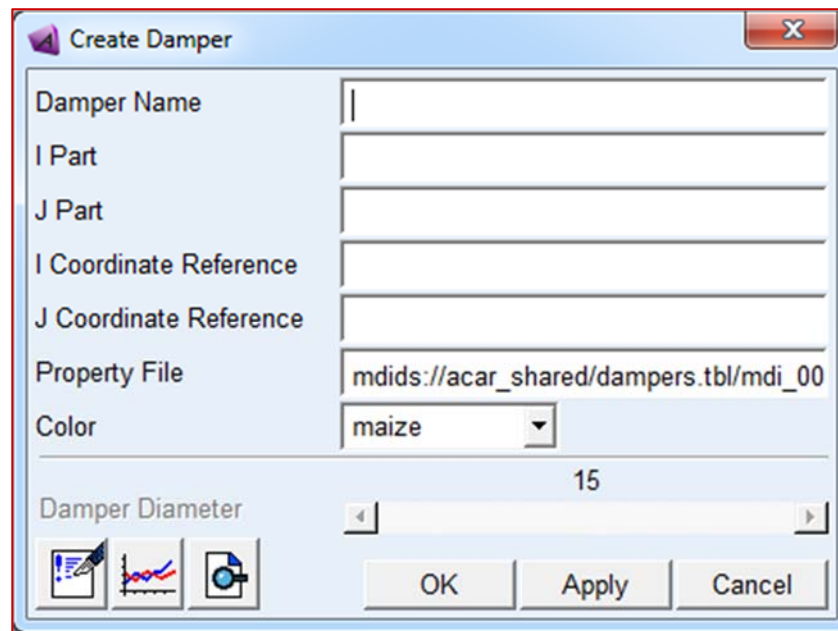


Where,

- C** is a constant.
- FL** is the free length of the spring.
- IL** is the defined installed length.
- DM(i,j)\*** is the initial displacement between the I and J coordinate reference points.
- If you enter a smaller value for **DM(i,j)\***, Adams/Car calculates an increased preload for the spring; conversely, a decreased preload.
- Force** represents the spring force.
- K** is the nonlinear spring stiffness derived from the property file.

# DAMPERS

- When creating dampers, you specify the two endpoints and the property file.
- Unlike a spring, a damper doesn't need a preload.
- Adams/Car also creates the geometry for the damper, like the one shown.



# BUMPSTOPS AND REBOUNDSTOPS

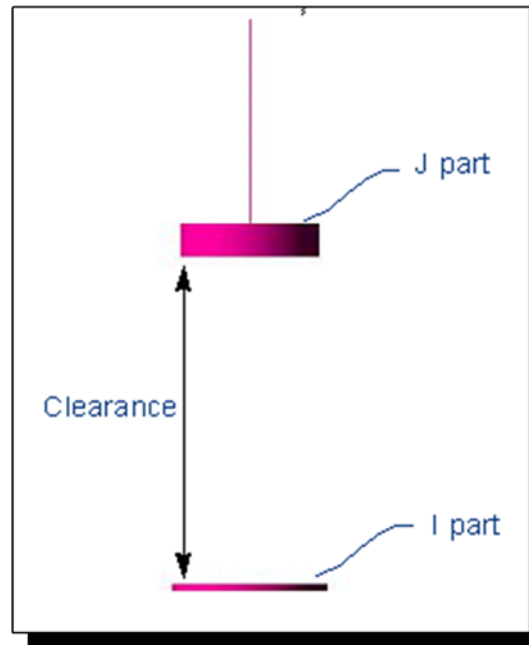
- **Bumpstops**

- Define a force-displacement relationship between two parts.
- Acts between user-specified coordinate reference points on each part, and conform to the force-displacement curve described in the designated property file.
- The bumpstop force is activated when the displacement between the two coordinate references exceeds a certain value, that can be defined by either of the following:
  - Clearance - Defines how much part I can travel towards part J before the force activates.
  - Impact length - Defines how close part I can come to part J before the force activates.

# BUMPSTOPS AND REBOUNDSTOPS

- **Reboundstops**

- Work similarly to bumpstops, but act in rebound, instead of jounce like bumpstops.
- Adams/Car also creates graphics for the elements shown next:



# TOE/CAMBER ANGLES AND SUSPENSION PARAMETER ARRAY

- You can set the toe and camber of a suspension by selecting **Build → Suspension Parameters → Toe/Camber Values → Set.**
- Here you set the design-position values of toe and camber, after which Adams/Car automatically creates parameter variables and output communicators, which you can then be used to orient the wheels or other entities with a construction frame.
- You need to create variables defining toe and camber angles.
- These are defined using a steer axis, which the suspension parameter array calculates.
- You access the suspension parameter array dialog box by selecting **Build → Suspension Parameters → Characteristic Array → Set.**



# TOE/CAMBER ANGLES AND SUSPENSION PARAMETER ARRAY

- **You can create a steering axis using either of these methods:**
  - Geometric method- Pick two points on different parts that define the steer axis.
  - Instant-axis method - Adams/Car first locks the spring travel and applies an incremental steering torque or force. Then, from the resulting translation and rotation of the wheel carrier parts, Adams/Car calculates the instant axis of rotation for each wheel carrier. The instant axes of rotation are the steer axes.

The screenshot shows the 'Suspension Parameters Array' dialog box. The 'Steer Axis Calculation' dropdown is set to 'Geometric'. The 'Suspension Type' dropdown is set to 'Independent'. A red rectangular box highlights the input fields for 'I Part', 'J Part', 'I Coordinate Reference', and 'J Coordinate Reference'. At the bottom, there are 'OK', 'Apply', and 'Cancel' buttons.

The screenshot shows the 'Suspension Parameters Array' dialog box. The 'Steer Axis Calculation' dropdown is set to 'Instant Axis'. The 'Suspension Type' dropdown is set to 'Independent'. A red rectangular box highlights the input fields for 'Part' and 'Coordinate Reference'. At the bottom, there are 'OK', 'Apply', and 'Cancel' buttons.

# ADJUSTABLE FORCES

- **An adjustable force is a special Template Builder user-defined element (UDE). You can use adjustable forces to satisfy static parameters in your model**
- **The current formulation creates a single-component force that acts between the two parts.**
- **You access the adjustable force functionality from Build → Adjustable Force → New.**
- **A typical application is using an adjustable force to set toe and camber values during a static suspension analysis:**
  - You might use two parts to define the tie rod and attach them by a translational joint.
  - You would then apply an adjustable force between the two parts to set toe and camber values.
  - When the vehicle reaches static equilibrium without the use of adjustable forces, the toe and camber alignments might not be the ones that you want.

# PARAMETER VARIABLES

- **A parameter variable is used to store key information in the template. For example:**
  - Toe/camber angle for a suspension
  - Maximum steering angle
- **Parameter variables have the naming convention `pv[lrs]_(name)`.**
- **Parameter variables can also be hidden from standard users. Hidden parameter variables cannot be modified through Standard Interface. The naming convention for these variables is `ph[lrs]_(name)`.**
- **Parameter variables can also store text.**

# GENERAL ADVICE

- **Make a sketch on paper first.**
- **Start with the hardpoints.**
- **Follow corporate naming or numbering convention where possible. This ensures that your templates can be successfully used by others within your company.**
- **Proceed down through the Build menu.**
- **Write comment strings when creating new objects.**
- **For some components, you have more options in the modify dialog box than in the create dialog box (for example, rigid part). In that case, create the component first and then parameterize it properly using the modify dialog box.**

# EXERCISE

- **Perform Workshop 12, “TEMPLATE-BUILDER TUTORIAL”**





# **SECTION 13**

## **COMMUNICATORS**

# COMMUNICATORS

- **This section introduces communicators, which control how subsystems exchange information in an assembly.**



# COMMUNICATORS

- **What's in this section:**
  - Types of Communicators
  - Classes of Communicators
  - Communicator Symmetry
  - Communicator Roles
  - Naming Communicators
  - Matching Communicators During Assembly
  - Matching Communicators with Test Rigs

# TYPES OF COMMUNICATORS

- **Communicators exchange information between different subsystems as well as between the subsystems and the testrig.**
- **A communicator is an Adams/Car variable.**
- **A communicator contains either a(n):**
  - Object (for example, a part, variable, marker, or joint)
  - Real value (for example, x,y,z location)
  - String

# TYPES OF COMMUNICATORS

- **Types of Communicators**

- Adams/Car has two types of communicators:

- **Input communicators** - Demand information from other subsystems or test rigs.
    - **Output communicators** - Provide information to other subsystems or test rigs.

- For example:

- In the wheel template, you have an input communicator for the wheel center location named **ci[lr]\_wheel\_center**.
    - In the suspension template you have an output communicator for the wheel center location named **co[lr]\_wheel\_center**.
    - When assembling these subsystems Adams/Car provides information to the input communicator from the output communicator if the names match.

# CLASSES OF COMMUNICATORS

- **The class of a communicator indicates the kind of information it exchanges. For example, communicators of the class hardpoint exchange a location through a hardpoint name and a part name.**
- **The classes of communicators and the information that each class exchanges are listed in the table on the following page. The classes apply to both input and output communicators.**

# CLASSES OF COMMUNICATORS

The class:	Exchanges:
Mount	Part name to provide connections between subassemblies. As a shortcut, Adams/Car also automatically creates input mount communicators when you create a mount part.
Location	The location of the named hardpoint or construction frame. If the hardpoint is part of a symmetrical pair, Adams/Car creates two input communicators, one for each hardpoint in the pair.
Marker	Construction frame and part name to provide location, orientation, and part information. If the construction frame is part of a symmetrical pair, Adams/Car creates an input communicator for each hardpoint in the pair.
Joint	Joint name.
Joint-for-motion	Joint name.
Bushing	Bushing name.
Array	Adams/Solver array name.
Spline	Spline name.

# CLASSES OF COMMUNICATORS

Differential equation	Differential equation name.
Solver variable	Adams/Solver variable name. You must use an Adams/Solver variable and not an Adams/Car variable. Unlike an Adams/Car variable, an Adams/Solver variable's computation occurs during analysis. Adams/Car generates Adams/Solver variables as state variables.
Motion	Motion name.
Part	Part name.
Orientation	The orientation of the named construction frame.
Real parameter	A parameter variable name of the type real.
Integer parameter	A parameter variable name of the type integer.
Force	Force name

*Note:* In the dialog boxes, the terms *class* and *entity* are interchangeable.

# COMMUNICATOR SYMMETRY

- A communicator can be either single or be part of a symmetrical pair, either left or right.
- Entity classes (array, differential equation, motion, parameter variable, solver variable, and spline) have no symmetry and, therefore, are always single, by default.

**Note:** The symmetry of the input communicator created automatically by a mount part is inherited from the coordinate reference.

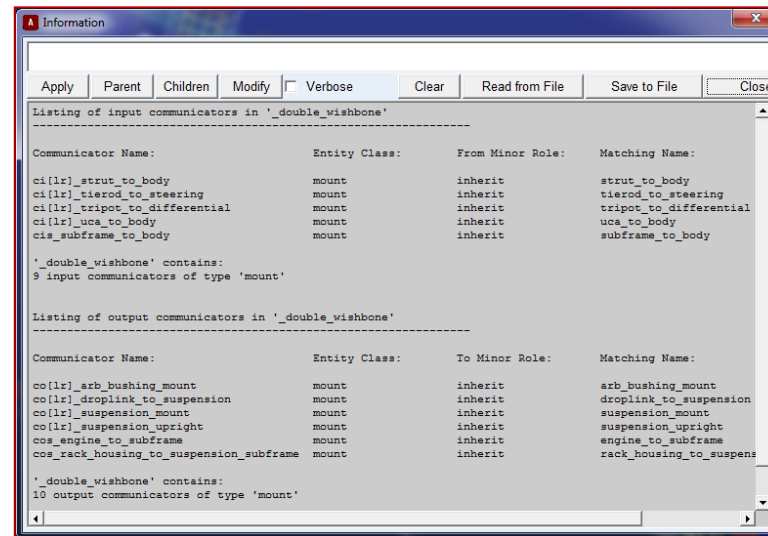
# COMMUNICATOR ROLES

- **Each communicator has a minor role. Adams/Car provides you with five default minor roles:**
  - Front
  - Rear
  - Trailer
  - Inherit
  - Any
  - Custom minor roles can be defined using the acarBS.cmd
- **The minor role is used in communicator matching logic.**
- **If you select inherit, the minor role of the communicator will become that of the subsystem using the template.**



# NAMING COMMUNICATORS

- After you create a communicator, Adams/Car assigns a prefix to the name.
- For example, it creates a prefix, *ci/* where: *ci* indicates it is an input communicator.
- If it were an output communicator, Adams/Car would use *co/* indicates it is for the left side of a symmetrical pair.
- If it were for the right side, Adams/Car would use an *r* (*cir*). If it were a single communicator, it would have an *s* (*cis*).



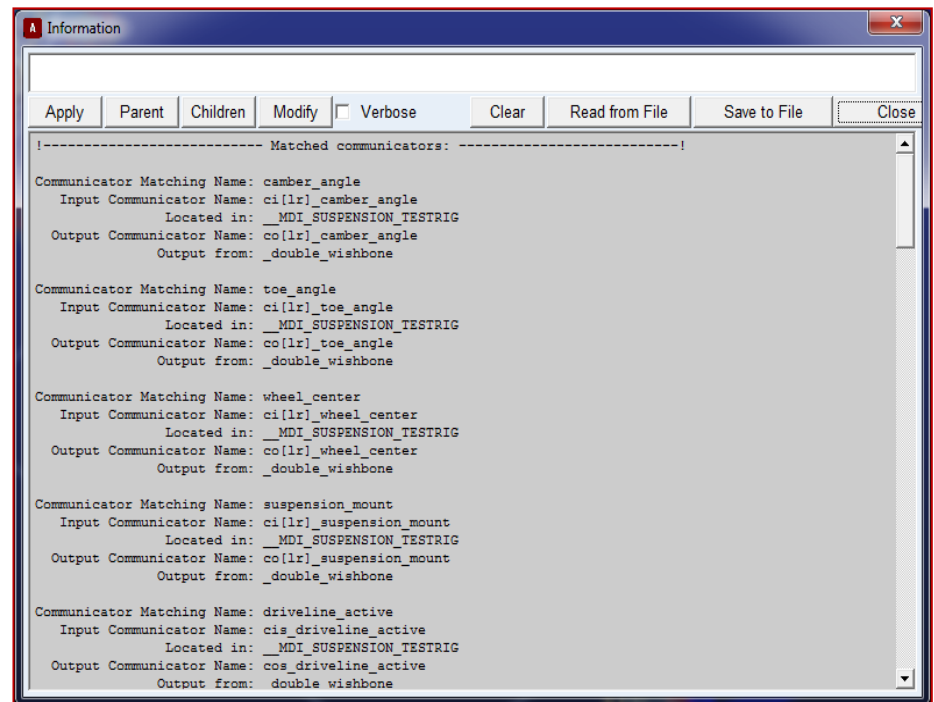
# NAMING COMMUNICATORS

- It uses the name of the mount part as the name of the communicator and appends the prefix *ci[lrs]\_* to it, depending on whether or not it is a left, right, or single communicator.
  - For example, if you create a mount part of *mtl\_rack\_mount*, Adams/Car creates an input communicator with the name *cil\_rack\_mount*, where *l* indicates it is for the left side.

**Note:** You cannot create a mount input communicator by itself. You must create a mount part, and Adams/Car will automatically create the communicator for you.

# MATCHING COMMUNICATORS DURING ASSEMBLY

- For a pair of communicators to exchange information during assembly, the communicators must:
  - Have same matching names.
  - Be of opposite types (one input, one output).
  - Be of the same symmetry type (left, right, or single).
  - Be of the same class (exchange the same type of information); for example, mount.
  - Have the same minor role or be assigned a role of any.



# MATCHING COMMUNICATORS DURING ASSEMBLY

- **Unmatched input communicators:**
  - If an input communicator does not have a corresponding output communicator, Adams/Car returns a warning message, and, if the input communicator belongs to the class *mount*, Adams/Car assigns the mount part to ground.
  - This warning message lets you know that your assembly may not be connected properly.

# MATCHING COMMUNICATORS DURING ASSEMBLY

- **Unmatched output communicators:**
  - If an output communicator is not linked up with one or more input communicators, you will not get a warning upon assembling your subsystems, because simply publishing information has no direct effect on the operation of your assembly.
  - You can still analyze the model if it does not have matching communicators. In fact, you may find this helpful if you want to run an analysis of a subsystem without attaching another subsystem to it.

# MATCHING COMMUNICATORS DURING ASSEMBLY

- Communicator matching example:

The pair:	Belongs to the class:	From minor role:	To minor role:
cil Strut_mount	mount	front	
col Strut_mount	mount		front
cil Strut_mount	mount	any	
col Strut_mount	mount		front
cil Strut_mount	mount	front	
col Strut_mount	mount		any

Diagram annotations:

- 1: Points to the first row of the table.
- 2: Points to the first two rows of the table.
- 3: Points to the first column of the table.
- 4: Points to the 'mount' entries in the second and third rows of the 'Belongs to the class' column.
- 5: Points to the 'front' entry in the first row of the 'From minor role' column and the 'front' entry in the second row of the 'To minor role' column.

# MATCHING COMMUNICATORS DURING ASSEMBLY

- **Additional rules:**

- An input communicator can only be matched with one output communicator.
- Output communicators can be matched with an unlimited number of input communicators.
- You should always check the warning messages during the assembly, especially if the warnings refer to an input communicator of class *mount* that does not get assigned and is, therefore, attached to ground.

# MATCHING COMMUNICATORS WITH TEST RIGS

- Test rigs require that certain communicators exist in the templates.
- For example, the suspension test rig requires the following communicators (among others):

The communicator:	Belongs to the class:	From minor role:	Matching name:
co[lr]_suspension_mount	mount	inherit	suspension_mount
co[lr]_suspension_upright	mount	inherit	suspension_upright
co[lr]_wheel_center	location	inherit	wheel_center
co[lr]_toe_angle	parameter_real	inherit	toe_center
co[lr]_camber_angle	parameter_real	inherit	camber_angle

- For a complete listing of communicators required by the test rigs, see the Templates tab in the Adams/Car online help.



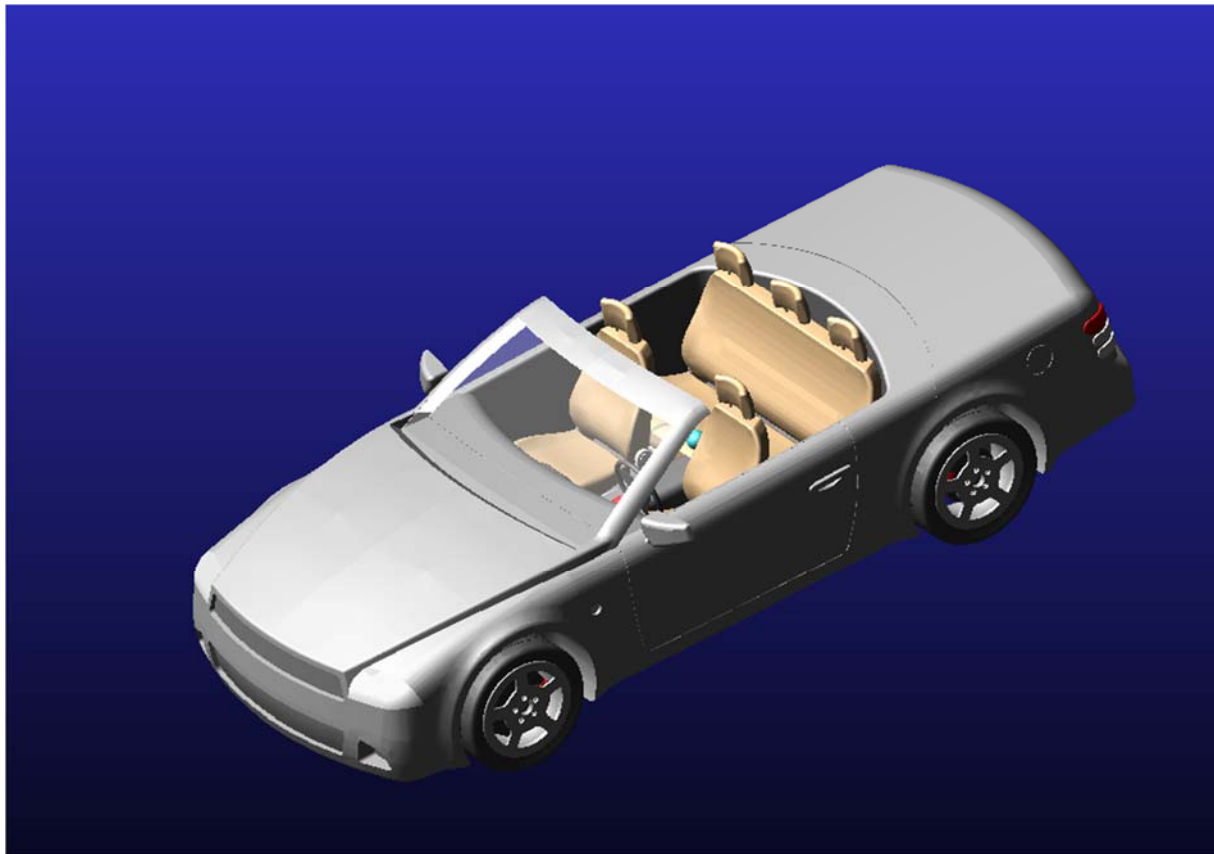
# EXERCISE

- **Perform Workshop 13, “GETTING INFORMATION ABOUT COMMUNICATORS”.**



# SECTION 14

## REQUESTS



# REQUESTS

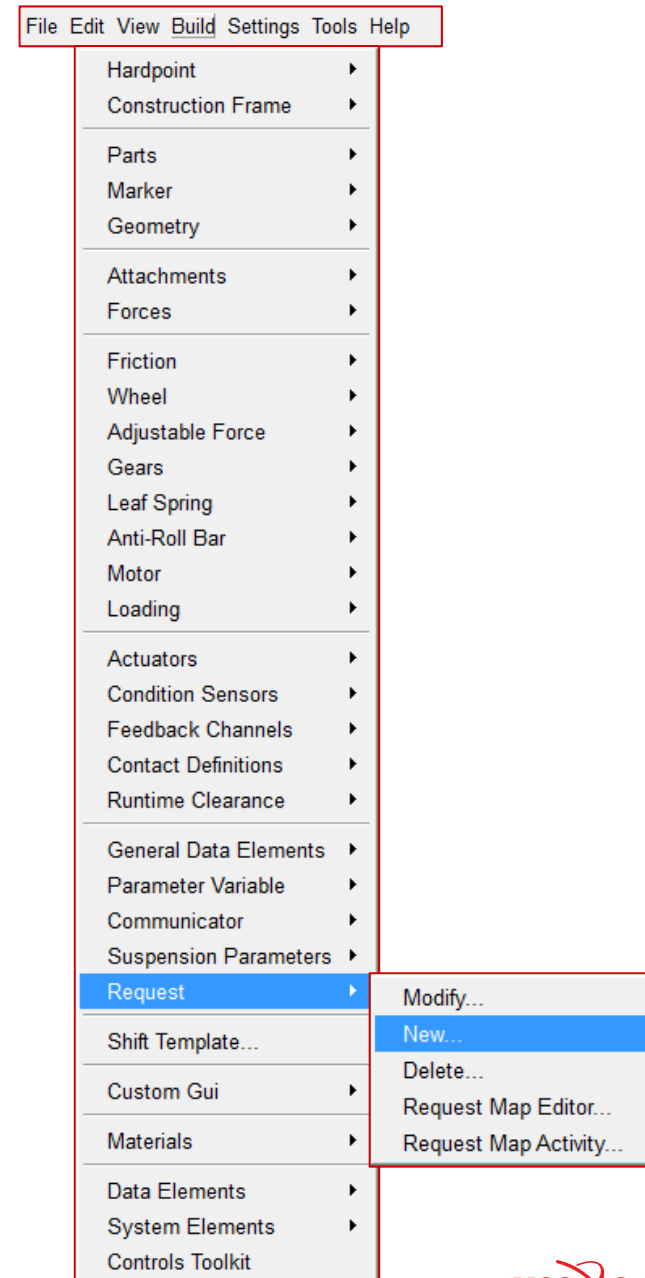
- **This section introduces requests, which are the primary method of output in Adams/Car.**

# REQUESTS

- **What's in this section:**
  - Creating New Requests
  - Types of Requests
  - Toggle Request Activity

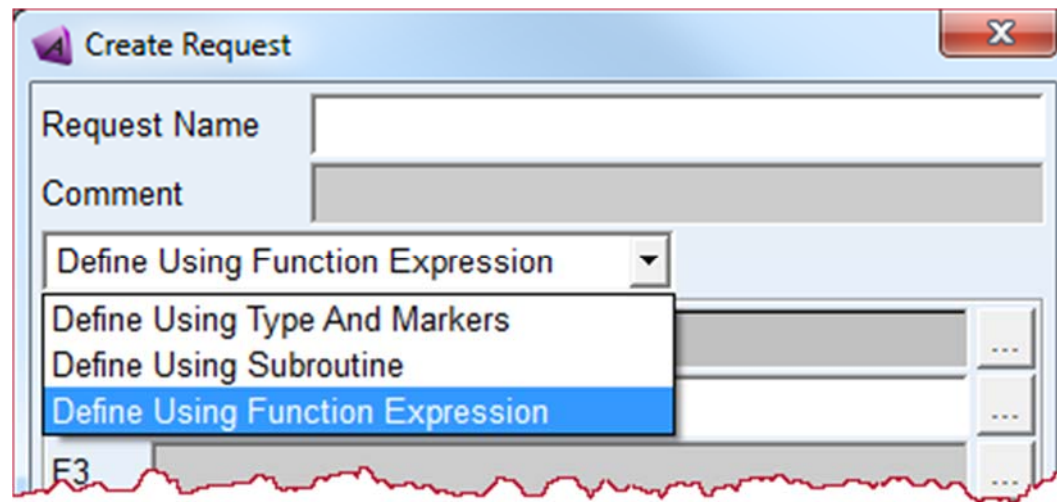
# REQUESTS

- **Creating New Requests**
  - Requests are created at the template level
  - Go to Build menu
  - Point to Request and select New



# TYPES OF REQUESTS

- **There are three types of requests that you can create**
  - Define Using Type And Markers
  - Define Using Function Expression
  - Define Using Subroutine



# TYPES OF REQUESTS

## 1. Define Using Type And Markers

- Measures displacement, velocity, acceleration and force/torque between two markers.
- You need to specify a To (I) marker, a From (J) marker and a Reference Frame (R) marker used for the computation.

## 2. Define Using Function Expression

- You need to explicitly specify the function used in the definition of the request.
- Function Expression examples include SFORCE, GFORCE, VZ, VM, DZ, DM



# TYPES OF REQUESTS

## 3. Define Using Subroutine

- You could write your own REQSUB subroutines for custom computation.
- Adams/Car uses many subroutines for request outputs. These use an internal numbering scheme. The example below is calling subroutine 905.

### Example:

There is a request named `"._double_wishbone.bkl_lca_front.force_request"` defined by the User Function shown below :

```
905.0, 3.0, (._double_wishbone.bkl_lca_front.i_marker[1].adams_id),  
           (._double_wishbone.bkl_lca_front.j_marker[1].adams_id),  
           (._double_wishbone.bkl_lca_front.field.adams_id)
```

# TYPES OF REQUESTS

## 3. Define Using Subroutine (Cont.)

– Where:

905: Request Number

3.0 : is the Force Request (you can have 1.0 and 2.0 as displacement and velocity requests respectively)

Similarly, the remaining components denote the I Marker , J Marker and Field respectively.

Using this request you can obtain the output components e.g. Forces and Torques.


This is how the request appears in the ADM dataset

```
=====
adams_view_name='TR_Rear_Suspension_bkl_lca_front_force_request'
REQUEST/94
FUNCTION = USER (905, 3, 282, 365, 29)
=====
```

# TOGGLE REQUEST ACTIVITY

- The menu pick “Tools → REQUEST Activity” in Standard Interface lets you activate/deactivate requests by certain criteria.
- Any activity change will only be retained during the current session because this setting is not saved in the subsystem (.sub) file.
- **Types of requests to activate/deactivate include:**
  - actuators
  - bushings
  - springs
  - dampers
  - bumpstops
  - reboundstops
  - all (of the above)
- To store the request activity at the subsystem level you can use special groups as shown on the following slide.

# TOGGLE REQUEST ACTIVITY

- **To store the request activity in Template Builder:**
  - Create a parameter variable to store the activity of the request as a 1 for active, 0 for inactive.
  - For example, pvs\_request\_activity.
  - Create a group with the command language: **Tools → Command Navigator**. Double click group and select create.
  - For more information on groups, press F1. 
  - Add your request names to this group.
  - See next slide to tie it together.

# TOGGLE REQUEST ACTIVITY

- **For the `expr_active` parameter of the group (1 = active, 0 = inactive), create a function that uses the parameter variable created on the previous slide.**

The commands for this might look similar to the following:

```
group create &  
  group_name = ._my_template.my_request_activity  
  objects_in_group = ._my_template.request0, &  
    ._my_template.request1, &  
    ._my_template.request2, &  
    ._my_template.request3, &  
  expr_active = (pvs_request_activity)
```

For more examples, investigate templates in the shared database which have the group `kinematic_mode_active`, which is used for the Kinematic Mode toggle.

# EXERCISE

- **Perform Workshop 14, “REQUESTS”.**



# **SECTION 15**

## **EXPLORING TEMPLATES**

# EXPLORING TEMPLATES

- **Before you build or modify templates, it is important that you know how templates are organized.**
- **Templates must describe their physical system: for example how parts are connected with joints, motions, and forces. Templates must also define the input/output communicators.**
- **This chapter introduces methods of exploring existing templates. This information will help you when modifying or creating templates.**



# EXPLORING TEMPLATES

- **What's in this section:**
  - Investigating Templates
  - Understanding Templates
  - About the Database Navigator

# INVESTIGATING TEMPLATES




- **Investigating templates primarily involves:**
  - Getting information about the template components.
    - Once you understand what components make up a template and how they relate to each other, you can modify the components to make your template unique.
  - Getting information about communicators.
    - Understanding how the template works mechanically is not enough: you must also understand how it communicates with other templates and test rigs. This understanding will help you connect your templates correctly.

# INVESTIGATING TEMPLATES

- **Methods for template investigation:**

1. To get information on a single component, right-click the component and then select **Modify** or **Info**.
2. The **Build** menu in template builder allows you to query info on most modeling elements.
  - For example **Build → Adjustable Force → Info** displays the Entity Information dialog box from which you can generate a listing of component information.
  - In Standard Interface you can use the **Tools → Entity Info** menu picks.

# INVESTIGATING TEMPLATES

3. When requesting information for communicators, your template-based product displays the Communicators Info dialog box. Use the Communicator Info dialog box to list the communicators in different templates and test rigs.
4. For a visual representation of how parts are connected, select **Tools → Highlight Connectivity**.
5. To get a text file listing all the parts and the joints and parts to which they are connected, on the Status bar, right-click  and then select either  or .
6. Use the Database Navigator to explore your template.

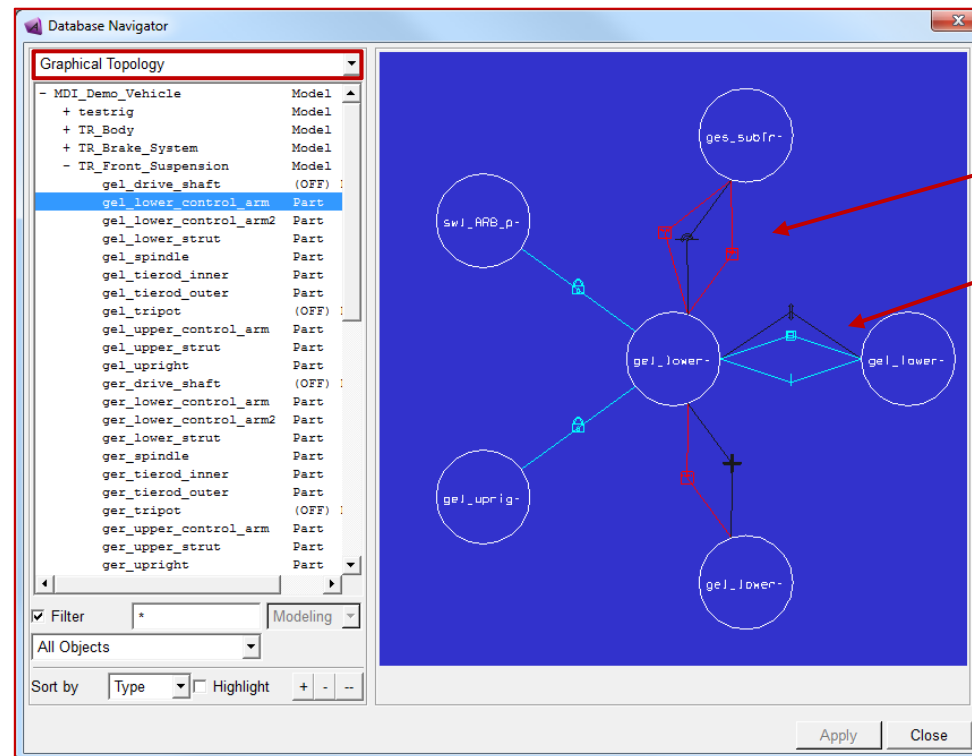
# ABOUT THE DATABASE NAVIGATOR

- **Displaying the Database Navigator**

- You can display the Database Navigator by doing any of the following:
  - From the **Tools** menu, select **Database Navigator**.
  - From the **Edit** menu, execute an editing command, such as **Modify**, when no object is currently selected.
  - From the **Edit** pop-up menu, request to view information about an object using the Info command.
  - Using the **Browse** command, browse for the name of an object to enter in a dialog box.

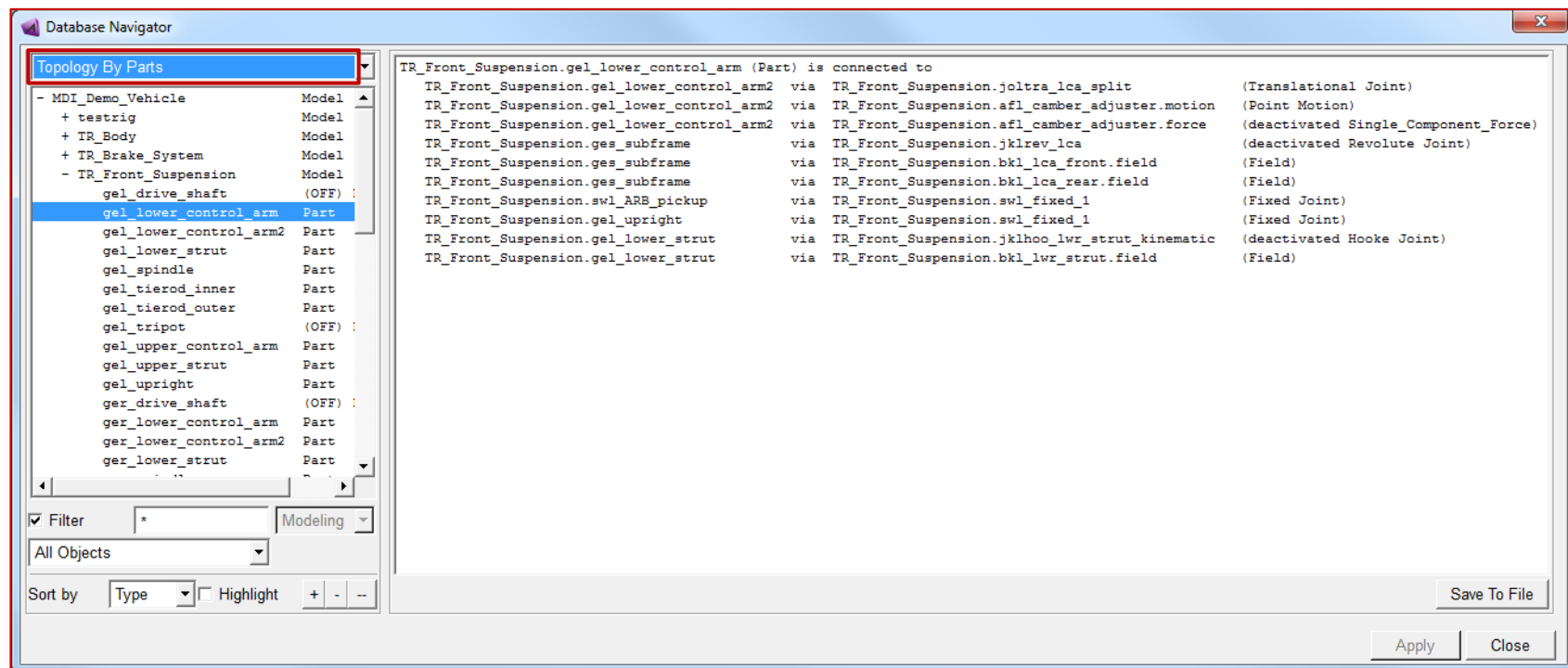
# ABOUT THE DATABASE NAVIGATOR

- To graphically view the topology of parts:
  1. From the option menu at the top of the dialog box, select **Graphical Topology**.
  2. From the tree list or view window, select an object.
  3. A graphical display of the object's topology appears in the text box to the right.



# ABOUT THE DATABASE NAVIGATOR

- **To view the topology of parts:**
  1. From the option menu at the top of the dialog box, select **Topology by Parts** or **Topology by Connections**.
  2. From the tree list or view window, select an object.
- **The topology of the object appears in the text box to the right.**



# ABOUT THE DATABASE NAVIGATOR

- **Viewing the associativity of objects**
  - You can use the Database Navigator to display the objects that a selected object uses. For example, you can select a joint in the tree list to show the I and J markers that the joint uses. You can also select to view the objects that use the selected object.
- **To view the associativity of objects:**
  1. From the option menu at the top of the dialog box, select **Associativity**.
  2. Set the associativity:
    - To show the objects that the selected object uses, select **Uses**.
    - To show the objects that use the selected object, select **Is Used By**.
  3. From the tree list or view window, select an object.
    - The objects associated with the selected object appear in the text box to the right.



# EXERCISE

- **Perform Workshop 15, “EXPLORING AND COMPLETING TEMPLATES”.**



# SECTION 16

## ADDITIONAL APPLICATIONS

# ADDITIONAL APPLICATIONS

- **Other applications offered by MSC Software are compatible with Adams/Car to provide additional analysis capabilities. The following describes how these applications fit in and affect the vehicle simulations in Adams/Car.**
- **These additional applications are incorporated into Adams/Car in the form of Plugins.**
- **For more information, refer to the documentation for each application. Most of these applications provide a “Getting Started” guide to give an overview of how they work.**

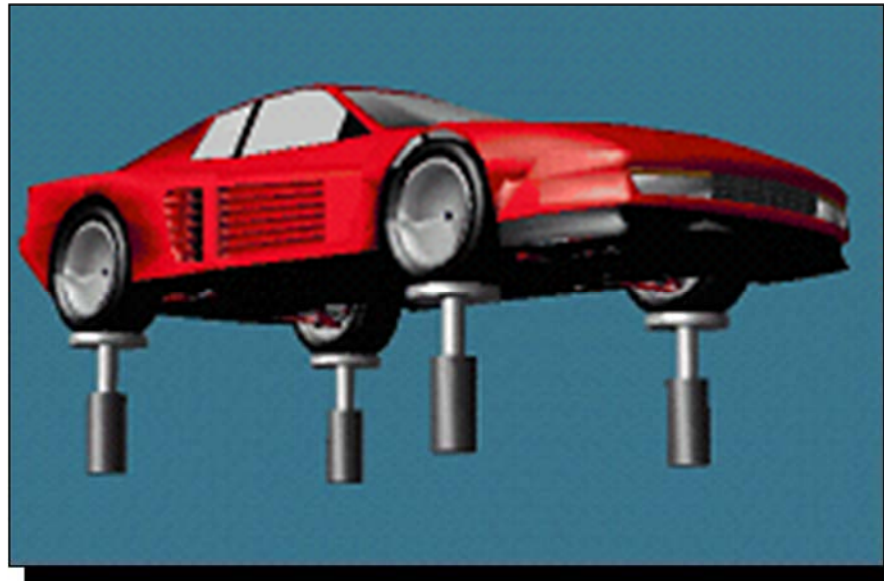
# ADDITIONAL APPLICATIONS

- **What's in this section:**

- Adams/Car Ride
- Adams/Vibration
- Adams/Durability
- Adams/Driveline
- Adams/Linear
- Adams/Controls
- Adams/Insight

# ADAMS/CAR RIDE

- **Adams/Car Ride is a plugin for Adams/Car that elevates the Functional Digital Vehicle solution from handling to ride and comfort testing, including the required elements, models, and event definitions for building, testing, and postprocessing within the ride-frequency domain.**



# ADAMS/CAR RIDE

- **The Adams/Car Ride user scenario is based on the existence of a vehicle virtual prototype.**
- **The same model database used for handling, is used for ride and comfort engineering.**
- **Adams/Car Ride enables you to exchange the mathematical description of the components (dampers, bushings, hydromounts, tires) so that the level of fidelity is suitable for the phenomenon you want to study.**

# ADAMS/CAR RIDE

- **The library of detailed components includes:**
  - A general, **GSE-based damper user-defined element (UDE) component**, is suitable to include a damper model generated from a detailed damper model description through the use of the Mathworks RealTime Workshop capability. For example, a simulink model is delivered with Adams/Car Ride to be used with this damper UDE.
  - A **detailed frequency and amplitude-dependent hydromount model**, whose mathematical description has been provided by Audi AG, Germany. This component model is completed with a stand-alone identification tool, which automatically identifies the parameters of the hydromount mathematical model so that its response matches user-specified characteristics of dynamic stiffness and loss angle.
  - A **detailed frequency-dependent bushing** model.
  - A **generic-component test rig** that allows you to validate the behavior of the components independently from the complete vehicle assembly. The component test rig allows excitation of from one to six input channels (three translations and three rotations), using either prescribed motion or prescribed force.



# ADAMS/CAR RIDE

- **Once you specify the components in detail, you can run your full-vehicle assembly through a battery of ride and comfort predefined simulation scenarios, enabling you to estimate typical system-level ride metrics.**
- **This is based on an extended four-post test rig, which supports analytical (for example, swept sine) as well as measured (for example, in RPC format) excitation inputs.**
- **This also include a road-roughness model to generate road profiles in terms of user-specified power spectral density (PSD).**
- **For all simulation events, Adams/Car Ride supports the time-domain and the frequency-domain approach.**

# ADAMS/VIBRATION

- **Adams/Vibration is an Adams plugin for performing frequency-domain analyses. Using Adams/Vibration, you can study forced vibrations within your Adams models.**
- **For example, you can**
  - simulate driving an automobile over a bumpy road
  - measure its frequency response
  - Both inputs and outputs are described in the frequency domain.

# ADAMS/VIBRATION

- **Using Adams/Vibration, you can:**
  - Analyze the forced **response of a model in the frequency domain** over different operating points.
  - Include the effects of hydraulics, controls, and user-subsystems in the frequency analysis.
  - **Transfer your linearized model** from Adams products to Adams/Vibration completely and quickly.
  - Create input and output channels for vibration analyses.
  - Specify **frequency domain input functions**, such as swept sine amplitude/frequency, PSD, and rotational imbalance.
  - Create user-defined, frequency-based forces.

# ADAMS/VIBRATION

- Solve for system modes over frequency range of interest.
- Evaluate frequency response functions for magnitude and phase characteristics.
- Animate forced response and individual mode response.
- Tabulate system modal contributions to forced vibration response.
- Tabulate contribution of model elements to kinetic, static, and dissipative energy distribution in system modes.
- Specify direct kinematic inputs.

# ADAMS/DURABILITY

- **Adams/Durability enables you to work faster and smarter, letting you easily interface with durability test machines using the RPC III file format, and with fatigue life calculation programs using DAC file format, or with an FEA program for stress recovery.**
- **Adams/Durability provides the following benefits:**
  - Reduces disk space requirements and improves performance by providing direct file input and output in RPC III and DAC formats.
    - For example, when you perform a 25-second Adams simulation with 300 channels of data, sampled at a rate of 409.6 points per second, to capture a durability event, the Adams request files are approximately 48 MB, whereas the RPC III file is only six MB.

# ADAMS/DURABILITY

- **Provides access to the system-level simulation capabilities of Adams/View, or vertical products, such as Adams/Car, and Adams/Driveline.**
- **Provides access to dynamic stress recovery methods using MSC Nastran or ANSYS.**
- **Performs modal stress recovery of flexible bodies.**
- **Performs quasi-static stress recovery of rigid bodies.**
- **Provides access to component life prediction using MSC Fatigue or FE-Fatigue.**

# ADAMS/DRIVELINE

- **Adams/Driveline allows you to build and test functional virtual prototypes of complete drivelines or driveline components.**
- **Adams/Driveline is implemented either as a plugin to Adams/Car or standalone, has the same look and feel as Adams/Car, and allows you to perform the same kinds of analyses.**
- **Using Adams/Driveline**
  - You can use the virtual prototypes created with Adams/Driveline for **performance analyses, component fatigue life estimation and NVH characteristics.**
  - You can also import those virtual prototypes into Adams/Car to study full-vehicle dynamics with a driveline included.

# ADAMS/DRIVELINE

- **Using Adams/Driveline**

- Similar to Adams/Car, Adams/Driveline has two operational modes:
  - **Standard Interface** – Allows designers and analysts to build up driveline designs from a database of available templates, fill in the design-specific numerical values, and run tests.
  - **Template Builder** – Allows experienced users to modify templates, or create new ones, to accommodate specific corporate needs.
- Adams/Driveline template libraries provide a wide range of components, including clutches, flexible shafts, synchronized gear pairs, bearings, viscous couplings, differentials, backlash, and gears.
- The Adams/Driveline catalog of tests and analyses includes split mu, uphill and downhill driving, impulse, step, and ramp torques, tip-in tip-out, gear shift, rock cycle, and clutch misuse.



# ADAMS/DRIVELINE

- **Using Adams/Driveline**

- Application areas for Adams/Driveline include:

- Conceptual tool for driveline layout
    - Full-vehicle handling analysis in front-wheel drive, rear-wheel drive, and all-wheel drive configurations
    - Torque transfer and torque distribution studies
    - Vibration analysis at the system level and the component level
    - Component fatigue life prediction
    - Control system design and verification
    - Driveline packaging studies
    - Studies of secondary motion at high operating speeds
    - Studies of the influence of flexible components

# ADAMS/LINEAR

- **Adams/Linear**

- Adams/Linear's key function is to **linearize nonlinear Adams equations**. The resulting set of equations enables you to calculate **the natural frequencies (eigenvalues) and mode shapes (eigenvectors)** associated with your mechanical design.
- Adams/Linear also helps bridge the gap between **control systems design and Adams mechanical simulation capabilities** by allowing you to generate state space matrices (the plant model), which relate the measured outputs of the system to the controlled input.

# ADAMS/CONTROLS

- **Adams/Controls**

- Control algorithms can now be used in the vehicle model.
  - For example, including automatic controls in the suspension and handling design of a vehicle makes it possible to avoid an obstacle on wet pavements.
- The control system designer can tune the controller for the particular vehicle design on the computer, with confidence that the Adams model provides accurate results.
- Adams/Controls allows the designer to see the effect of each change on the computer, even comparing control strategies that would be too expensive or time consuming to test on a physical prototype.
- This interface allows you to model control algorithms in Matlab or EASY5.

# ADAMS/INSIGHT

- **Adams/Insight**

- When using Adams/Insight together with Adams/Car, you can plan and run a series of experiments for measuring the performance of your suspension or vehicle.
- Adams/Insight will help you to understand your designs better by distinguishing between key and insignificant design parameters.
- In addition, you will be able to see the impact of design decisions on a broad range of customer requirements, as well as improving design robustness by considering manufacturing variations up front.
- To learn how to use Adams/Car with Adams/Insight, see the Examples section in the Adams/Car online help.



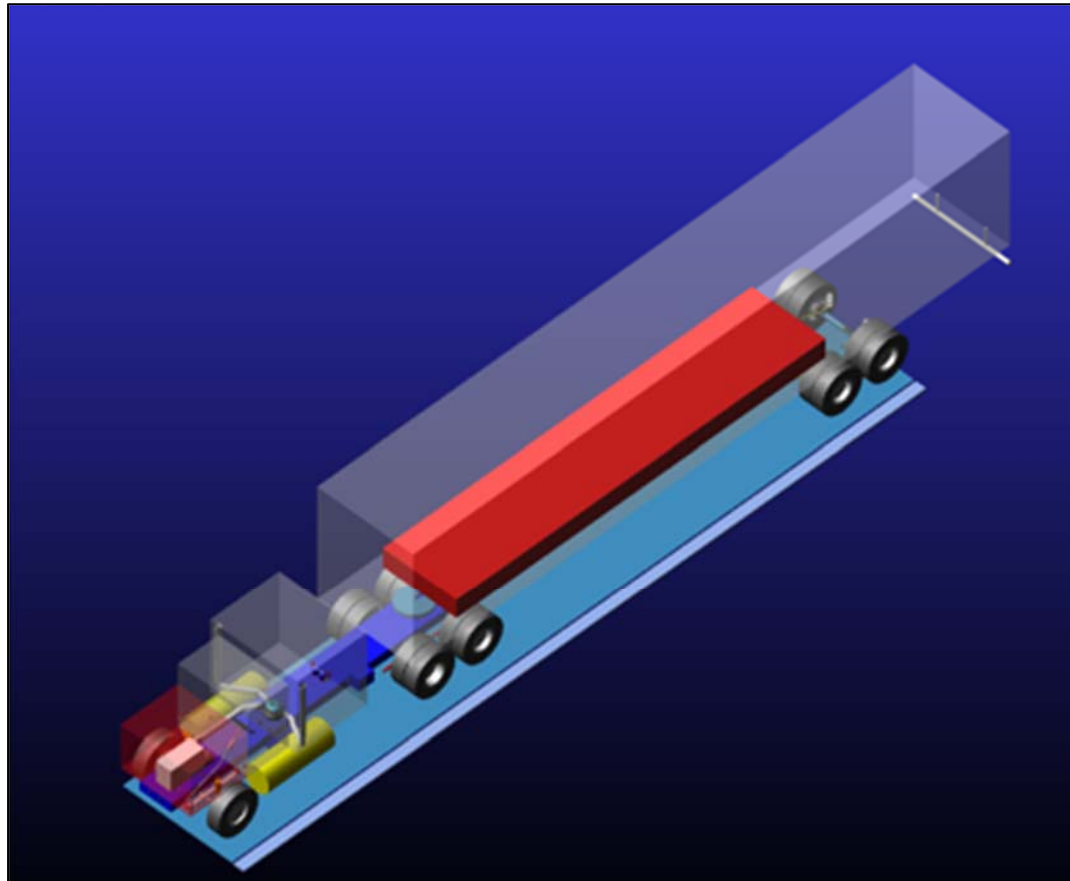
# EXERCISE

- **Perform Workshop 16, “LINEAR MODES ANALYSIS”.**



# SECTION 17

## TILT TABLE ANALYSIS



# TILT TABLE ANALYSIS

- **In the Adams/Car-Truck Plugin, a tilt table analysis capability is included to help engineers determine the stability property of vehicles.**



# TILT TABLE ANALYSIS

- **What's in this section:**
  - Tilt Table Overview
  - Adams/Truck Plugin
  - Setting up the analysis
  - Inputs to the analysis

# TILT TABLE OVERVIEW

- **Tilt table tests are commonly performed on vehicles to estimate the aggregate CG height and rollover point.**
- **This capability is supported for all Adams/Car full vehicle models provided that they are assembled with the tilt table test rig from the Truck plugin.**
- **The angle of inclination of the tilt table is increased either quasi-statically or dynamically and the simulation is terminated automatically (by a SENSOR) when tire vertical forces reach a user defined threshold value.**

# ADAMS/TRUCK PLUGIN

- The 'Adams/Car Truck' plugin must be loaded in order to access the tilt table testrig named `.__TILT_TABLE_TESTRIG`.
- The 'Adams/Car Truck' plugin also contains example models for a tractor semi-trailer as well as a bus.

# SETTING-UP THE ANALYSIS

- To perform a tilt test simulation, you first must build a full-vehicle assembly referencing the new testrig `.__TILT_TABLE_TESTRIG`.
- The testrig includes joint primitives at each of the downhill wheels to prevent the vehicle from sliding off the table. These joints attach to the wheels automatically by use of communicators.
- The testrig is set up to attach to wheels from subsystems with the following minor roles: front, rear, rear\_2, tag\_axle, trailer\_1, trailer\_2. In this way the same testrig may be used with vehicle assemblies containing from two to six axles.

# INPUTS TO THE ANALYSIS

- **The following inputs are required for a Tilt Table analysis:**
  - Tilt rate (angle/time): Tilt table's angular velocity.
  - Maximum Angle: The tilt angle at which to stop the simulation.
  - Force Sensor Threshold: The minimum tire force at which to stop the simulation.
  - Move Right Side: Specify the direction of tilt.

# EXERCISE

- **Perform Workshop 17, “TILT TABLE ANALYSIS”.**

# SECTION 18

## USING ADAMS/INSIGHT WITH ADAMS/CAR

# USING ADAMS/INSIGHT WITH ADAMS CAR

- **Within Adams/Car, Adams/Insight lets you design sophisticated experiments for measuring the performance of your suspension and full vehicle system.**



# USING ADAMS/INSIGHT WITH ADAMS CAR

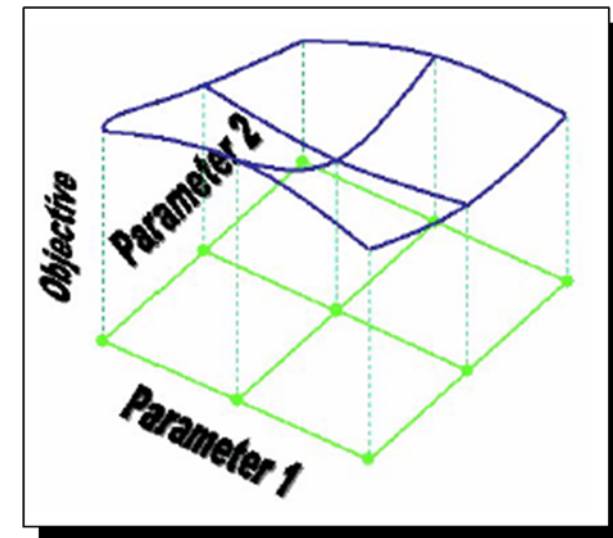
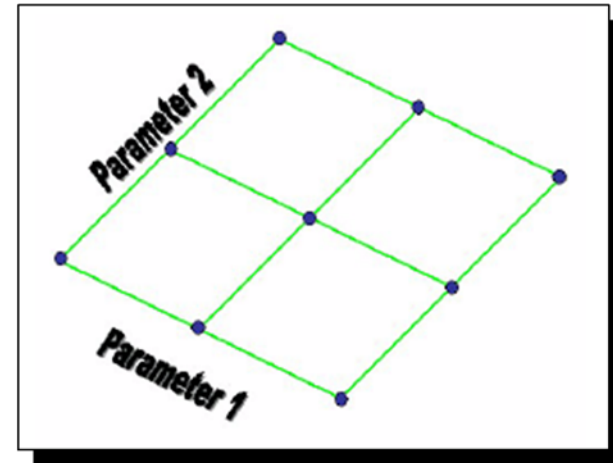
- **What's in this section:**
  - What is Experimental Design?
  - Response Surface Overview
  - Response Surface Equations
  - Factor (Parameter) Definition
  - Response (Objective) Definition
  - Typical Samples Required for a Cubic Fit
  - Response Surface Equation and Quality
  - Pareto Chart Output
  - Starting Adams/Insight from Adams/Car
  - Fitting & Publishing Results

# WHAT IS EXPERIMENTAL DESIGN?

- **Experimental design (also called Design of Experiments (DOE)) is a collection of procedures and statistical tools for planning experiments and analyzing the results.**
  - General DOE process:
    - Choose a set of factors (variables, or inputs) for the system that you are investigating and develop a way to measure the appropriate system responses (outputs, or objectives).
    - Specify a fit relationship (linear, quadratic, cubic) between the inputs and outputs. This determines the number of trials to run in the experiment.
    - Execute the runs, fit the response surface in Adams/Insight.
    - Analyze the response surface, asking questions such as:
      - Do the response surface statistics indicate a good fit?
      - Which factors most influence which objectives?
      - Can you optimize certain objectives using the fitted response surface?

# RESPONSE SURFACE OVERVIEW

- **DOEs for design space exploration:**
  - A DOE study breaks up the design space intelligently, sampling at enough locations in order to compute a best-fit between inputs and outputs.
  - Response surfaces fit to data:
    - Fit polynomial equation to the input, output values. This produces a 'response surface' that is continuous and analytical.
    - Any design configuration can be quickly predicted from the response surface equation. This makes for rapid optimizations, tolerance studies, etc.



# RESPONSE SURFACE EQUATIONS

- **Polynomial relationship sought between inputs and outputs:**
  - Linear:  $y = a + bx$
  - Quadratic:  $y = a + bx + cx^2$
  - Cubic:  $y = a + bx + cx^2 + dx^3$
- **User specifies relationship \*beforehand\*. Simulations are run at each sampling point and results are fitted at the end.**

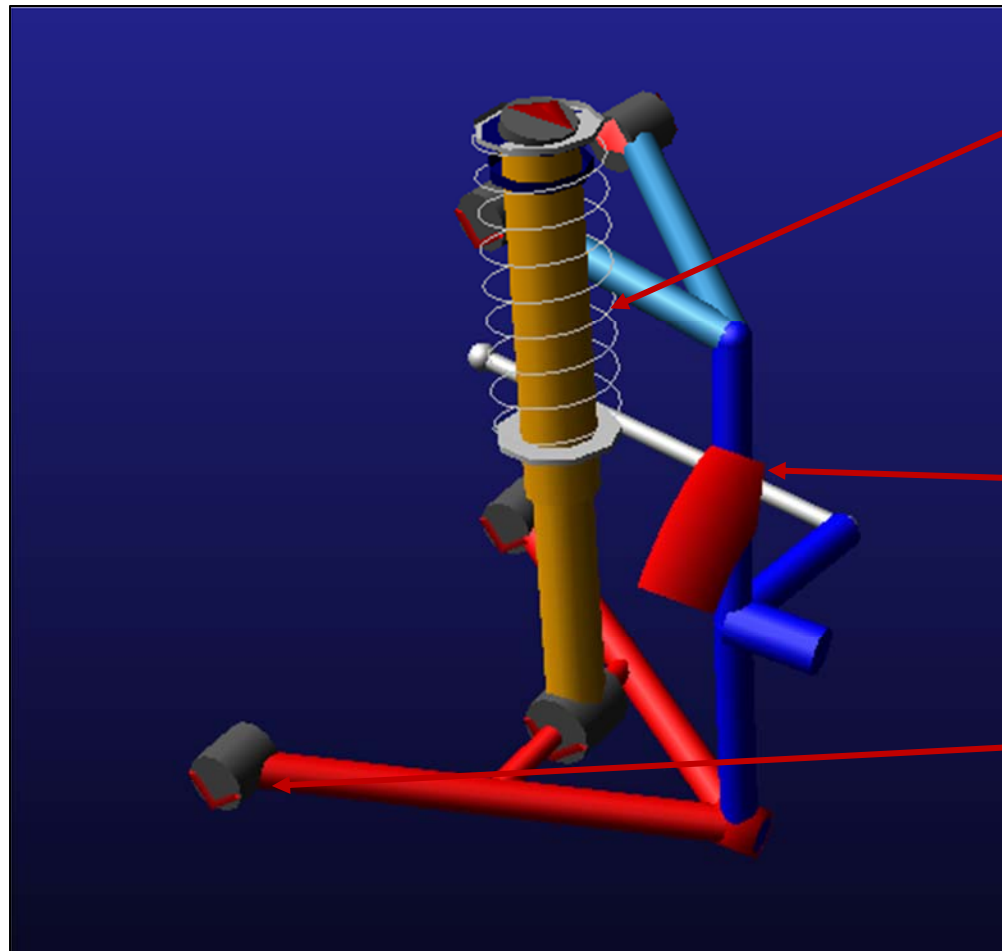
# FACTOR (PARAMETER) DEFINITION

- **Factors are the quantities being altered in Adams/Insight. Factors are defined by hardpoints and parameter variables, for example:**

Factor	Description
hp[lr]_lca_outer	Location of lower control arm attachment
pvs_front_brake_mu	Front brake coefficient of friction

- **Hardpoints appear in Adams/Insight with variable x, y, z quantities.**
- **Parameter variables need a range (high/low) defined in Adams/Insight.**

# EXAMPLE FACTORS



pvs\_spring\_k

pvs\_brake\_mu

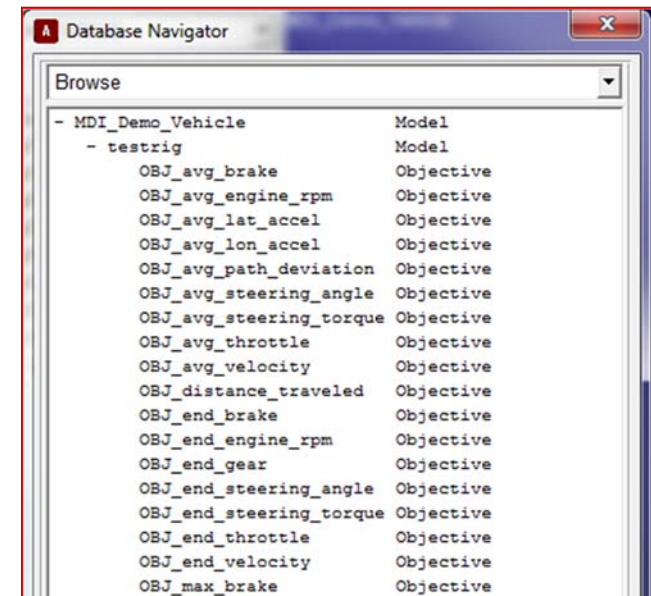
hp[lr]\_lca\_front

# RESPONSE (OBJECTIVE) DEFINITION

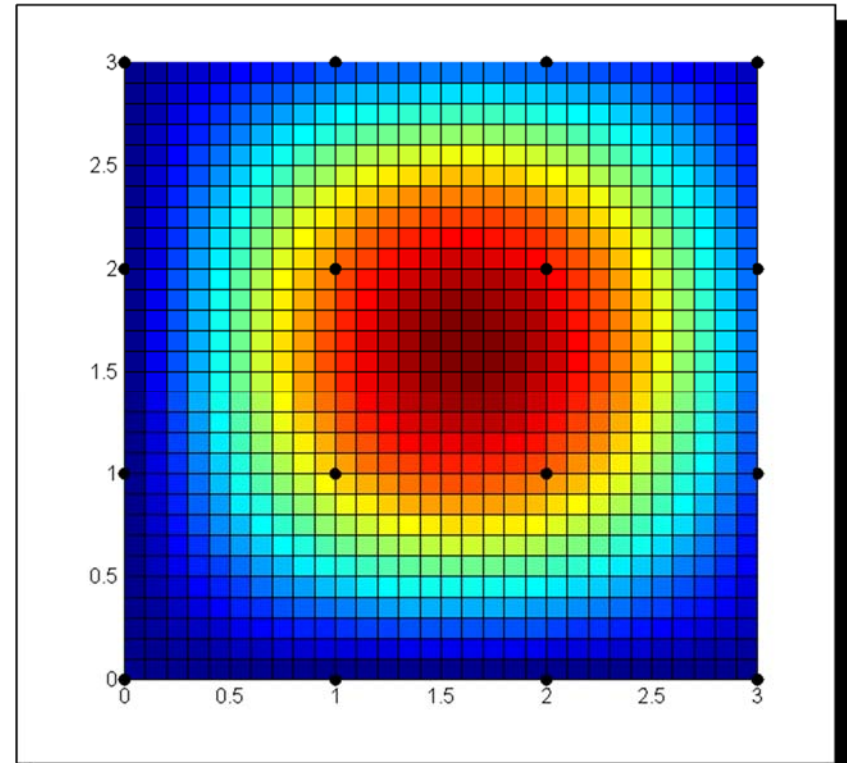
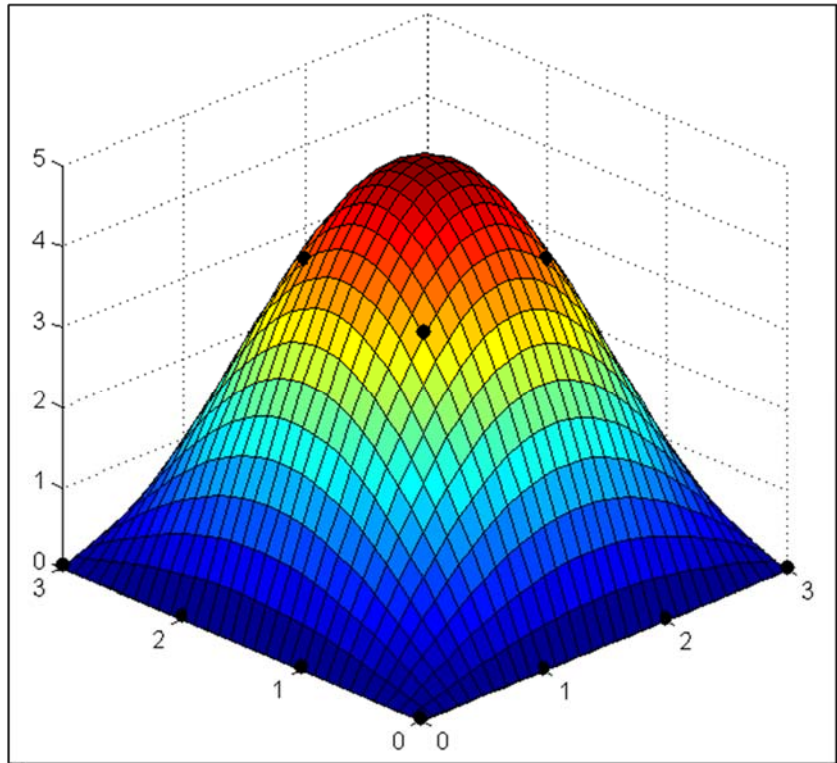
- Objectives in Adams must be a single quantity that is derived from a time history. These quantify the characteristic of interest, for example:

Objective	Description
OBJ_avg_lat_accel	Average lateral accel throughout event
OBJ_max_yaw_rate	Maximum yaw rate throughout event

- Common Objectives can be found in the testrig, as shown on the right.
- User-defined Objectives may be created at the Template level.



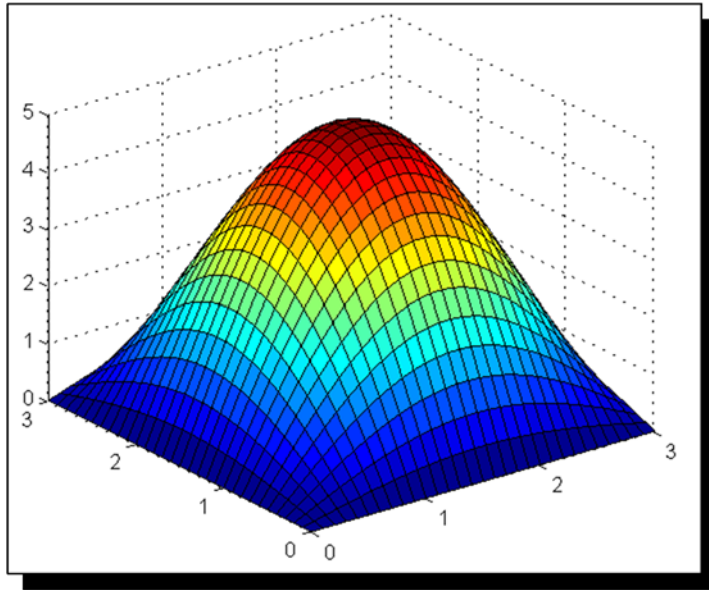
# TYPICAL SAMPLES REQUIRED FOR A CUBIC FIT



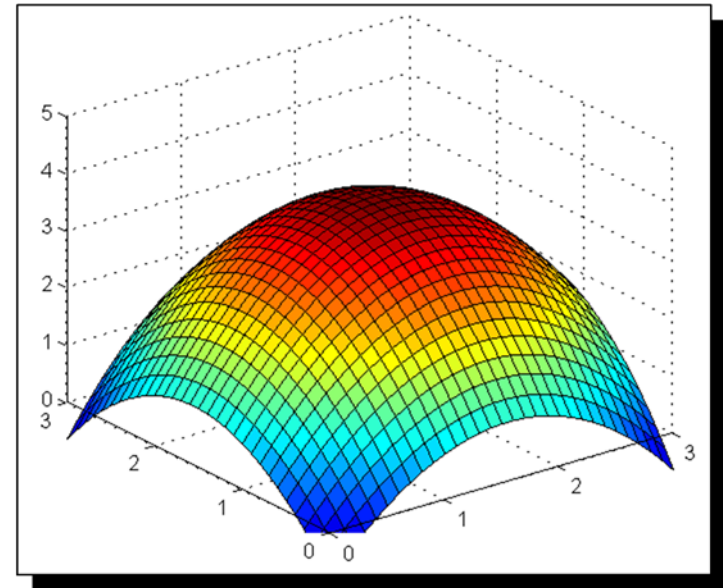
- **Cubic fit has 4 unknowns in each variable; 16 total unknowns for example above. 16 samples of design space required as shown.**



# COMPARE RESPONSE SURFACE TO ORIGINAL DATA

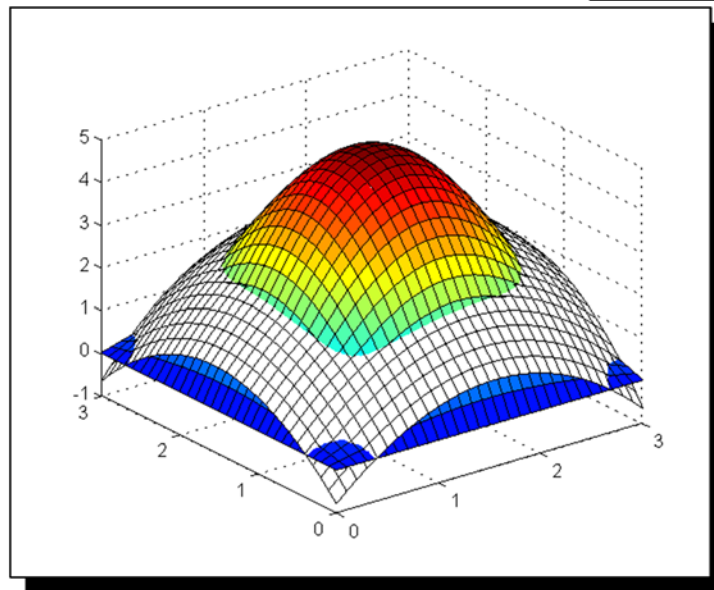


**Original**

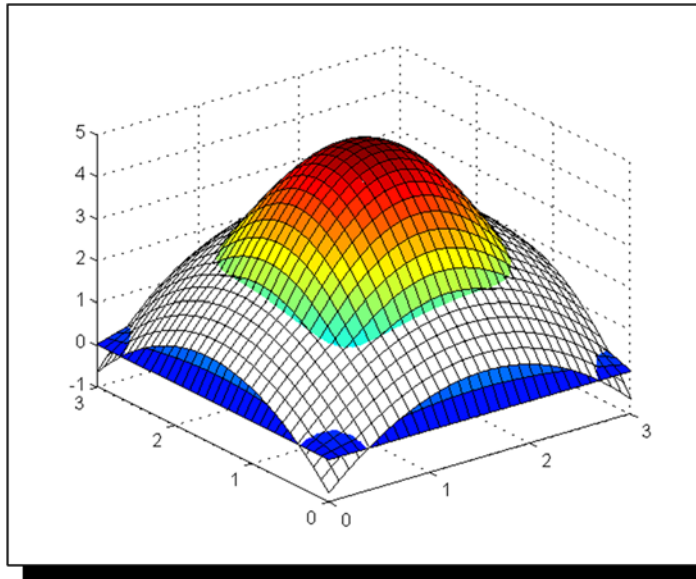


**Response  
Surface**

**Combined**



# RESPONSE SURFACE EQUATION AND QUALITY



Terms for regression "z"

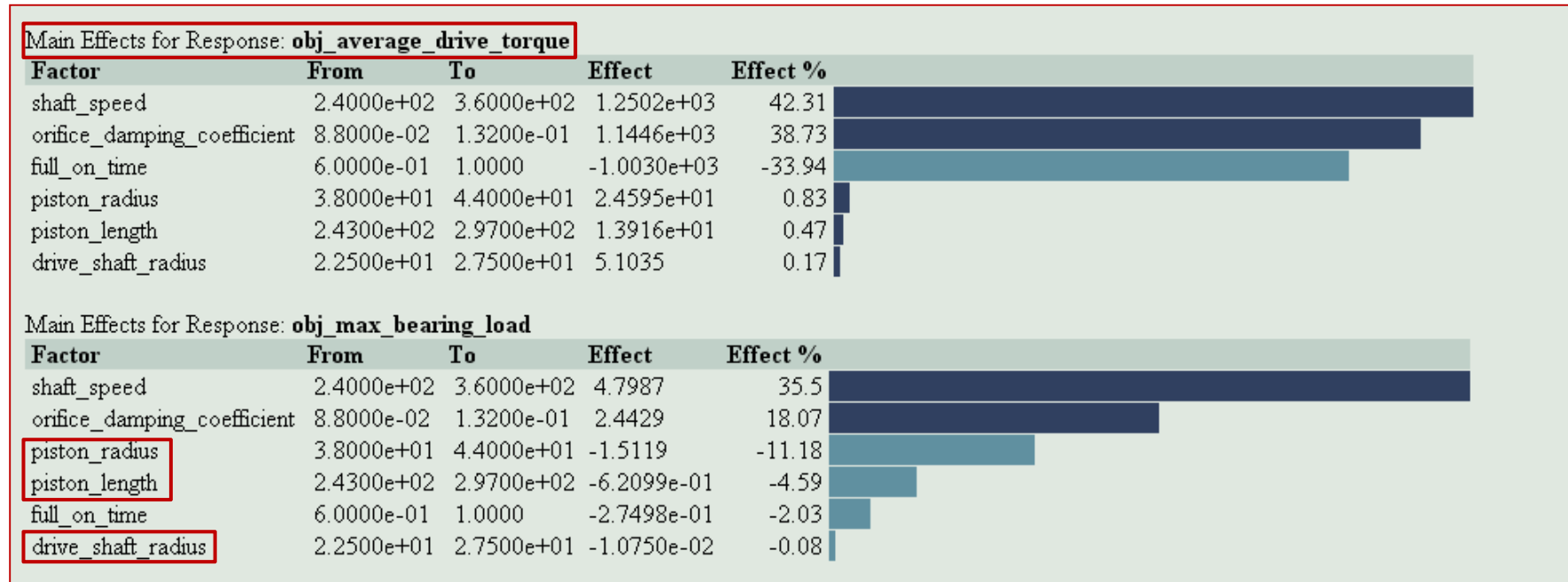
	Coef	+/-	Std. Error	Beta	T	P		Term
1	-0.79488	3.0099	1.2301	1.3058e-016	-0.6462	0.54206	?	1 (constant)
2	2.2939	7.1147	2.9076	1.6136	0.78893	0.46018	?	x
3	2.2939	7.1147	2.9076	1.6136	0.78893	0.46018	?	y
4	0.60504	3.1377	1.2823	1.0206	0.47184	0.65371	?	x * y
5	-0.6928	5.5937	2.286	-1.5256	-0.30306	0.77208	?	x^2
6	-0.6928	5.5937	2.286	-1.5256	-0.30306	0.77208	?	y^2
7	-0.09883	0.72366	0.29574	-0.47432	-0.33417	0.74962	?	x * y^2
8	-0.09883	0.72366	0.29574	-0.47432	-0.33417	0.74962	?	x^2 * y
9	-0.024579	1.2061	0.4929	-0.16763	-0.049865	0.96185	?	x^3
10	-0.024579	1.2061	0.4929	-0.16763	-0.049865	0.96185	?	y^3

$$Z_{fit} = -0.79 + 2.29x + 2.29y + 0.6x^2 - 0.69y^2 - 0.098xy^2 - 0.098x^2y - 0.02x^3 - 0.02y^3$$

- Adams/Insight provides the equation coefficients, the statistics of the fit and more.
- Here the visual fit is mediocre; the yellow icons reflect the statistics of this. Green/red icons indicate acceptable/poor fit quality.

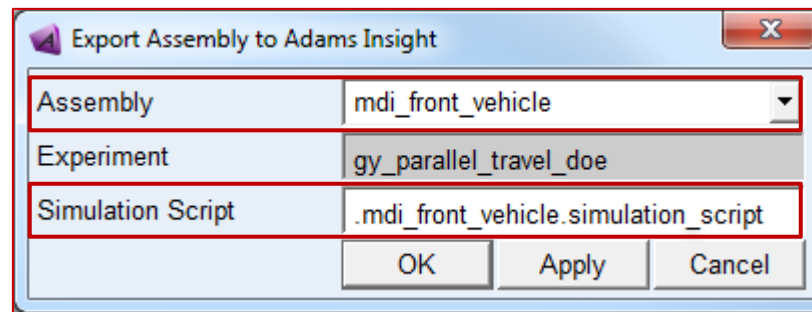
# PARETO CHART OUTPUT

- Adams/Insight can also produce Pareto charts which show the relative effect of a variable (factor) on an outcome (response).



# STARTING ADAMS/INSIGHT FROM ADAMS/CAR

- **To start Adams/Insight from Adams/Car:**
  - From the main menu in Adams/Car, point to Simulate, point to DOE Interface, point to Adams/Insight, and then select Export.
  - The Export Assembly to Adams/Insight dialog box appears.



- The steps on working inside Adams/Insight is detailed in the Adams/Insight online help.
- For more background on DOE, optimization and stochastics in Adams:  
<https://simcompanion.mscsoftware.com/infocenter/index?page=content&id=WM156>

# FITTING RESULTS

- Once Adams/Car has completed the trials defined in your work space matrix, you can use Adams/Insight to fit your results to a polynomial or a response surface.
- The purpose of fitting your results is to establish a relationship between the factors and responses that you selected for the work space matrix. Fitting results includes a multiple regression.
- You will be able to investigate the parts of the regression in the Summary, located in the TreeView under Analysis
- To fit your results:
  - From the **Adams/Insight** toolbar, select the **Fit results** tool . You can also select the **Tools** menu, and then select **Fit New Model**.
  - The Model Properties Summary window appears. Here, you can enter information on your model.

# PUBLISHING RESULTS

- **Adams/Insight lets you save your findings as either HTML or SYLK files.**
- **Once saved, you can use either a browser or spreadsheet program, such as Excel, to modify factors and see the effect on responses without performing full simulations.**
- **To publish your results:**
  - In the treeview, under Analysis, select Model, and then go to the **Adams/Insight** toolbar, and select the **Export to Web, SLK, etc.** tool .
  - You can also select the **File** menu, point to **Export**, and then select **Model**. In the window that opens, set the **File Type** to HTML File.

# EXERCISE

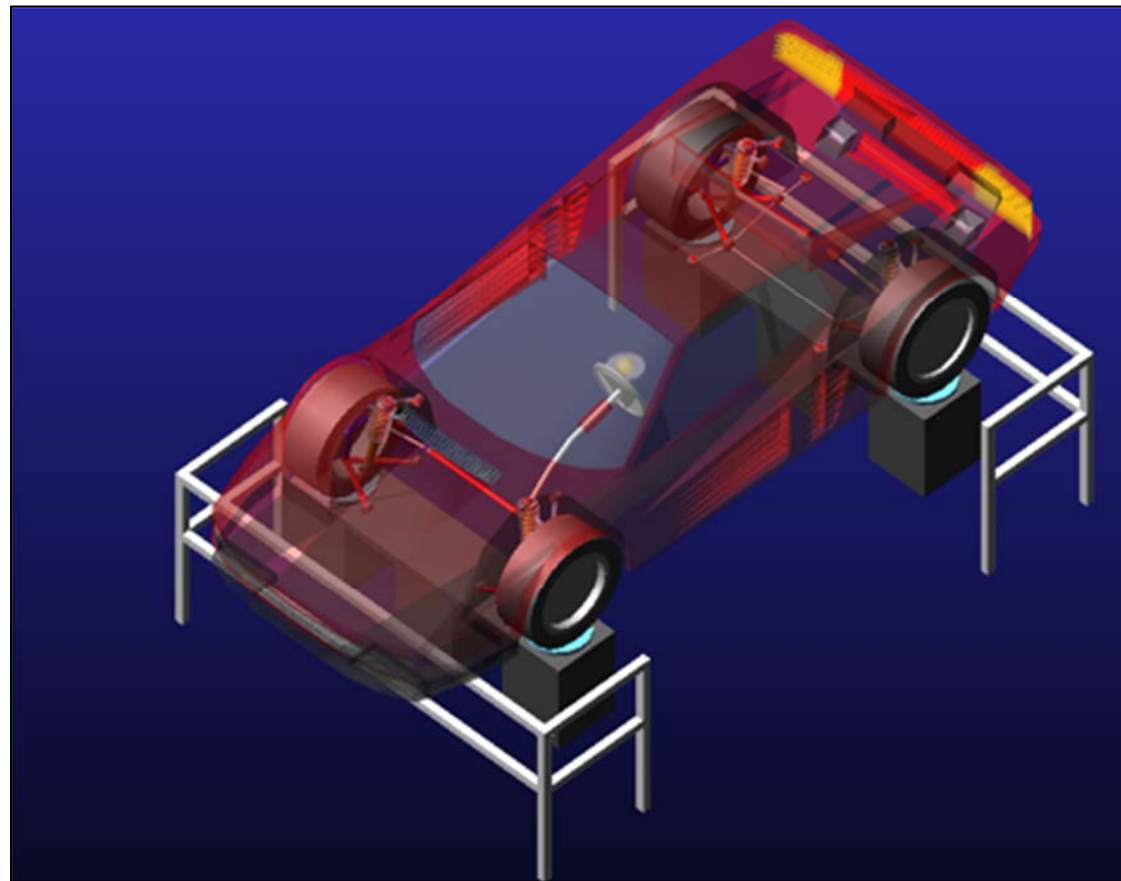
- **Perform Workshop 18, “USING ADAMS INSIGHT WITH ADAMS CAR”.**





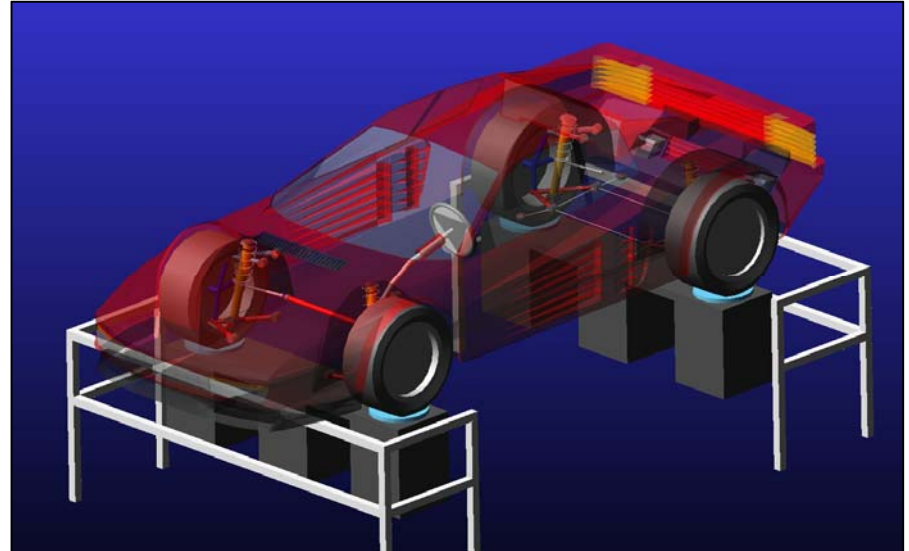
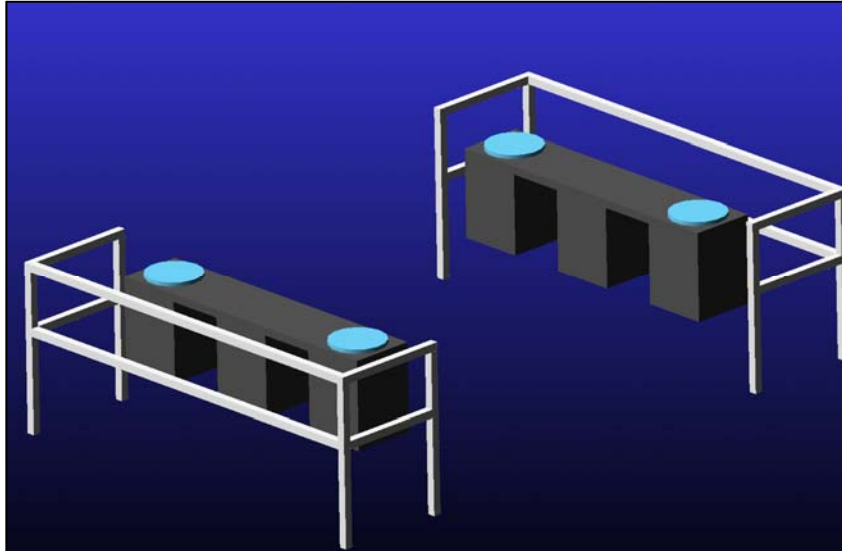
# SECTION 19

## SVC AND SPMM EVENTS IN ADAMS CAR



# SVC AND SPMM

- This section provides an overview of the Static Vehicle Characteristics (SVC) and Suspension Performance Measurement Machine (SPMM) events.



# SVC AND SPMM

- **What's in this section:**
  - Static Vehicle Characteristics overview
    - SVC overview
    - Suspension SVC
    - Suspension SVC output
    - Full vehicle SVC
    - Sprung mass table
    - Full vehicle SVC output
  - Suspension Performance Measurement Machine
    - SPMM overview
    - SPMM sub-events
    - SPMM testrig
    - SPMM output
    - SPMM output postprocessing

# SVC OVERVIEW

- **Static Vehicle Characteristics (SVC) are used to determine the characteristics of automobile or light truck suspensions at static equilibrium.**
- **Typical examples of such characteristics include wheel rate and front suspension percent anti-dive. SVC analyses can act on half- or full-vehicle models.**
- **For a full vehicle, SVC calculates characteristics for the front and rear suspension as well as general vehicle characteristics such as ground reactions and mass properties.**
- **For a half-vehicle, SVC computes only suspension characteristics.**

# SUSPENSION – SVC


- **Most of the Suspension SVC calculations are based on the compliance matrix for the vehicle's suspension. A CONSUB controls this analysis.**
- **Suspension geometry refers to the position and orientation of suspension parts relative to ground as the suspension is articulated through its ride, roll and steer motions.**
  - For example, the orientation of the spindle axes is used to compute the toe and camber angles.
- **There are two options to select the vertical mode:**
  - **Force:** Axle Load or total vertical force acting on the half-vehicle assembly
  - **Length:** You can define the vertical length of the center of the table for SVC analysis.

# SUSPENSION SVC OUTPUT

- Important information gathered from this simulation includes:
  - a. suspension rise
  - b. anti-lift/dive
  - c. wheel rates
  - d. suspension compliance
- SVC produces an ASCII report file summarizing the most useful suspension static vehicle characteristics calculated.

FRONT SUSPENSION CHARACTERISTICS				
Suspension Description: <acar_shared>/subsystems.tbl/MDI_FRONT_SUSPENSION.sub				
(PARAMETER)	(UNITS)	(AVERAGE)	(LEFT)	(RIGHT)
Unsprung mass (total)	kg	N/A		
Unsprung c.g. height	mm	N/A		
Roll center height	mm	144.84		
Wheel center rise	mm	21.92	21.92	21.92
Static loaded tire radius	mm	316.00	316.00	316.00
Track width	mm	1393.44		
Axle distance from vehicle cg	mm	N/A		
steer angle	DEG	0.00		
Toe Angle	DEG	356.9E-03	357.0E-03	356.8E-03
Caster Angle	DEG	-106.3E-03	-106.3E-03	-106.2E-03
Camber Angle	DEG	312.2E-03	312.1E-03	312.2E-03
Kingpin Angle	DEG	-311.5E-03	-311.5E-03	-311.5E-03
KP offset at wc, long	mm	40.47	40.47	40.47
KP offset at wc, lat	mm	74.75	74.75	-74.75
Caster Angle wrt body	DEG	-106.3E-03	-106.3E-03	-106.2E-03
Camber Angle wrt body	DEG	312.2E-03	312.1E-03	312.2E-03
Kingpin Angle wrt body	DEG	-311.5E-03	-311.5E-03	-311.5E-03
Scrub radius	mm	75.00	75.00	75.00
Caster trail	mm	-40.47	-40.47	-40.47
Toe change	DEG/mm	-16.26E-03	-16.27E-03	-16.26E-03
Caster change	DEG/mm	14.24E-03	14.24E-03	14.24E-03
Camber change	DEG/mm	-18.95E-03	-18.95E-03	-18.95E-03
Roll camber coefficient	DEG/DEG	768.3E-03	768.3E-03	768.3E-03
Percentage roll steer	%	19.23		
Track change	mm/mm		207.0E-03	207.0E-03
Wheelbase change	mm/mm	12.01E-03	12.01E-03	12.01E-03
Wheel rate	N/mm	45.50	45.50	45.50
Single bump wheel rate	N/mm	45.50	45.50	45.50
Ride rate	N/mm	37.07	37.07	37.07
Tire rate	N/mm	200.00	200.00	200.00
Roll rate - wheel	N mm/DEG	770.9E+03		
Roll rate - total	N mm/DEG	628.0E+03		
Wheel/spring ratio	mm/mm	960.2E-03	960.2E-03	960.2E-03
Wheel/shock ratio	mm/mm	960.2E-03	960.2E-03	960.2E-03

# FULL VEHICLE SVC

- This performs a Static Vehicle Characteristics for full vehicles.
- In addition to suspension characteristics, full vehicle SVC computes general characteristics such as ground reactions and mass properties (vehicle mass and inertia).
- SVC performs a calculation of the unsprung mass for the front and rear suspension. In order to make this calculation, each part must be assigned a sprung mass percentage.
- There is an option to enter either a single value for sprung mass % which is applied to all parts in front (or rear), or one can input the value of sprung mass % from each individual part.
  - *The sprung mass % value defaults to 100% if not specified*
- Upon selecting the option, single value, there is a table-like interface  that provides the user with an easy way of looking at a list of parts and the % sprung mass value and SVC array which can be toggled.



# SPRUNG MASS TABLE

- The Sprung Mass table can be used to change the sprung mass % value for the individual parts in the assembly.
- It is used to assign the parts of the subsystem with minor role 'any' to the front/rear svc array.
  - **Example:** Consider the case for all-wheel-drive powertrain, which may have minor role front/rear/any depending on where the major components are attached. In such a case one can add the parts in the desired array just by selecting the part and toggle it to the desired array type.

Sprung Mass Table

Assembly Subsystem MDI\_Demo\_Vehicle Filter Symmetry: Yes No Both

	Part Name	Mass	% Sprung Mass	SVC Array
1	TR_Front_Suspension.gel_lower_control_arm2	0.1	50.0	front
2	TR_Front_Suspension.gel_tierod_inner	0.3337368666	100.0	front
3	TR_Front_Suspension.gel_tierod_outer	0.3337368666	100.0	front
4	TR_Front_Suspension.gel_lower_control_arm	1.6113954942	50.0	front
5	TR_Front_Suspension.gel_upright	1.3972982748	0.0	front
6	TR_Front_Suspension.ges_subframe	25.0	100.0	front
7	TR_Front_Suspension.gel_upper_strut	1.0	100.0	front
8	TR_Front_Suspension.gel_upper_control_arm	1.0318710362	50.0	front
9	TR_Front_Suspension.gel_lower_strut	1.0	50.0	front
10	TR_Front_Suspension.gel_spindle	1.1028403931	0.0	front
11	TR_Rear_Suspension.gel_lower_control_arm2	0.1	50.0	rear
12	TR_Rear_Suspension.gel_tierod_inner	0.5	100.0	rear
13	TR_Rear_Suspension.gel_tierod_outer	0.5	0.0	rear
14	TR_Rear_Suspension.gel_lower_control_arm	1.6089138343	50.0	rear
15	TR_Rear_Suspension.gel_upright	1.3294519509	0.0	rear
16	TR_Rear_Suspension.ges_subframe	30.0	100.0	rear
17	TR_Rear_Suspension.gel_upper_strut	5.0	100.0	rear
18	TR_Rear_Suspension.gel_upper_control_arm	1.0318710362	50.0	rear
19	TR_Rear_Suspension.gel_lower_strut	5.0	50.0	rear
20	TR_Rear_Suspension.gel_spindle	1.1028403931	50.0	rear
21	TR_Rear_Suspension.gel_drive_shaft	4.2174529406	50.0	rear

Display: Single and Left Right Both Show SVC Array: Any and Front Rear Both

Toggle Selection To: Front Rear Any OK Apply Cancel

% Sprung Mass: current sprung mass percentage  
 SVC Array: indicates whether a part is to be included in the front or rear array.



# FULL VEHICLE SVC OUTPUT

- For a full vehicle, SVC calculates characteristics for:
  - a. front and rear suspension
  - b. general vehicle characteristics such as ground reactions for front and rear
  - c. Vehicle ( sprung) mass and inertia properties.

GENERAL CHARACTERISTICS				
(PARAMETER)	(UNITS)	(TOTAL)	(LEFT)	(RIGHT)
Total weight	N	14.98E+03		
Front ground reaction	N	6304.17	3159.20	3144.97
Rear ground reaction	N	8677.26	4345.82	4331.44
Total roll inertia	kg mm**2	298.7E+06		
Total pitch inertia	kg mm**2	1.162E+09		
Total yaw inertia	kg mm**2	1.340E+09		
Total product Ixy	kg mm**2	1.878E+06		
Total product Ixz	kg mm**2	4.950E+06		
Total product Iyz	kg mm**2	-391.2E+03		
Sprung mass	kg	1396.25		
Sprung roll inertia	kg mm**2	222.9E+06		
Sprung pitch inertia	kg mm**2	941.7E+06		
Sprung yaw inertia	kg mm**2	1.051E+09		
Sprung product Ixy	kg mm**2	1.890E+06		
Sprung product Ixz	kg mm**2	1.418E+06		
Sprung product Iyz	kg mm**2	-356.7E+03		
Total c.g. height	mm	401.63		
Sprung c.g. height	mm	409.26		
Body yaw angle	DEG	0.00		
Body pitch angle	DEG	-540.8E-03		
Body roll angle	DEG	16.34E-03		
Speed	mm/S	0.00		
Wheelbase	mm	2559.78	2559.79	2559.78
Bounce node loc. wrt H-pt.	mm	-4523.32		
Bounce natural frequency	Hz	1.50		
Pitch node loc. wrt H-pt.	mm	395.83		
Pitch natural frequency	Hz	2.44		
Ride frequency ratio		842.1E-03		

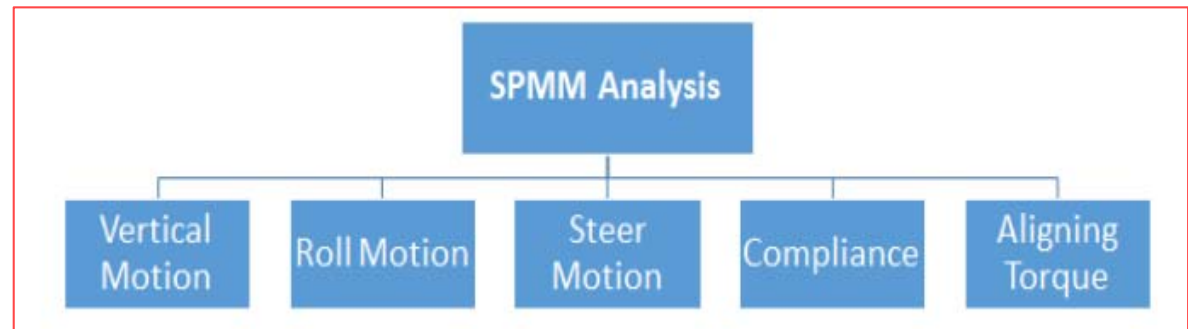
# SPMM

- **Suspension Parameter Measurement Machine (SPMM) is a Kinematics and Compliance (K&C) test machine.**
- **The K&C characteristics of a vehicle influence vehicle performance in terms of ride, steering and handling.**
- **Kinematic and Compliance (K&C) analyses quantify how the vehicle characteristics are altered by changes in:**
  - Suspension and steering system geometries,
  - Compliances due to suspension springs
  - Anti-roll bars
  - Elastomeric bushes and component deformations, etc.
- **SPMM applies known displacement, force and torque inputs. It measures the quasi-static suspension characteristics that are important to ride and handling.**

# SPMM SUB-EVENTS

- **The SPMM event has five different kinematic and compliance sub-events:**

- Vertical Motion
- Roll motion
- Steer Motion
- Compliance
- Aligning Torque



- **These sub-events evaluate parameters such as suspension rates, bump-steer, roll-steer, roll stiffness distribution, longitudinal and lateral compliance steer and steering system characteristics.**

# SPMM TESTRIG

- The **\_\_MDI\_SPMM\_TESTRIG** testrig is used for Suspension Parameter Measurement Machine (SPMM) analysis.
- This testrig actuates the chassis parts in jounce, rebound and roll motions. A GFORCE is used to apply forces and moments at the tire contact patch or at the wheel center for compliance and aligning events. The Steering sub-event actuates the steering wheel using a joint motion
- Before performing a SPMM analysis, you must specify several parameters about the vehicle in which you intend to use the suspension and steering subsystems. These parameters include the steering ratio, whether or not the suspension is front- or rear-wheel drive, and the braking ratio.

# SPMM OUTPUT

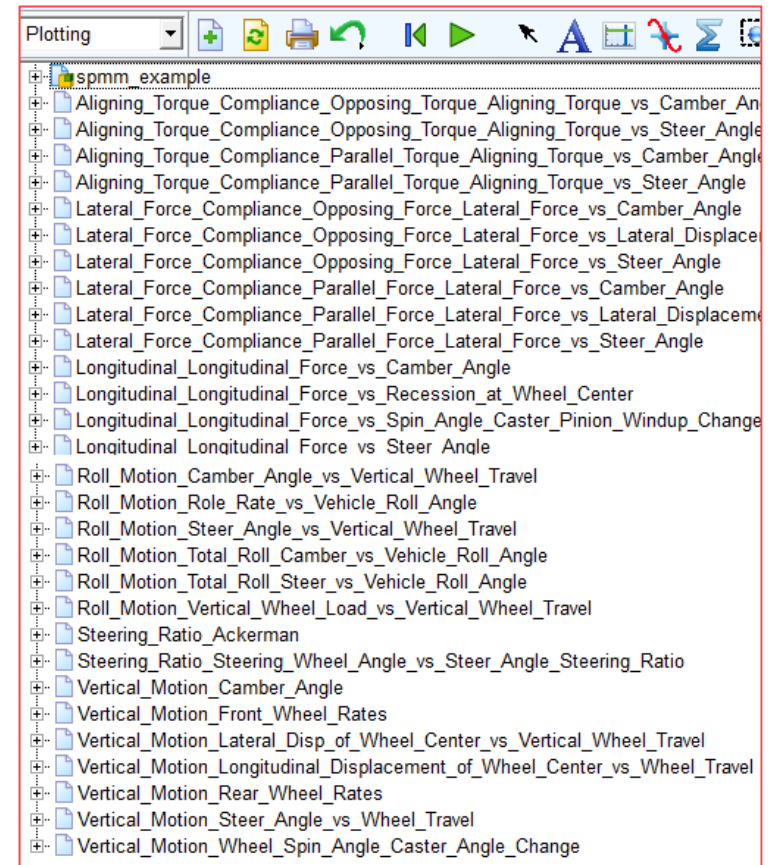
- The SPMM reports shows information at static condition related to vehicle K&C characteristics.
- SPMM event creates plot configuration files (.plt) in your working directory with combination of output prefix and sub-events name
  - (For example, ' <output\_prefix>\_vertical\_spmm.plt' for vertical motion sub-event).
  - It also creates command file (.cmd) that can be used to import all the plt files. The plot configuration file includes pre-defined plots for the selected SPMM sub-events.

SPMM TEST REPORT				
=====				
VEHICLE ASSEMBLY:				
<acar_shared>/assemblies.tbl/spmm_example.asy				
(PARAMETER)	(UNITS)	(TOTAL)	(LEFT)	(RIGHT)
Total weight	N	11882.14	5955.28	5926.86
	%		50.12	49.88
Front ground reaction	N [%]	6055.12 [51.0]	3034.89	3020.23
Rear ground reaction	N [%]	5827.02 [49.0]	2920.39	2906.63
Total roll inertia	kg-mm**2	2.8815131e+008		
Total pitch inertia	kg-mm**2	8.5565935e+008		
Total yaw inertia	kg-mm**2	1.0286609e+009		
Total product Ixy	kg-mm**2	1.3905086e+006		
Total product Ixz	kg-mm**2	5.6946956e+006		
Total product Iyz	kg-mm**2	-3.3818737e+005		
Global C.G location (X/Y/Z)	mm	1519.93	-1.91	441.14
Ground plane elevation	mm	16.22		
C.G. height	mm	424.92		
Body yaw angle	deg	0.000		
Body pitch angle	deg	0.45		
Body roll angle	deg	0.02		
Wheelbase	mm	2559.08	2559.09	2559.08
Average track width	mm	1556.88		
Static stability factor	-	1.832		
FRONT AXLE:				
<acar_shared>/subsystems.tbl/TR Front Suspension.sub				

# SPMM OUTPUT - POSTPROCESSING

- The plots can be created by importing \*\_plt.cmd file which is generated automatically after simulation and stored in your working directory.
- Individual results for sub-events is also available

Data	Math
Simulation	Request
test_vertical_spmm (<acar	displacement bgl_subframe_front_disp
test_roll_spmm (<acar_shar	velocity bgl_subframe_front_force
test_steer_spmm (<acar_sha	acceleration bgl_subframe_front_velo
test_long_cp_compliance_spmm	force bgl_subframe_rear_disp
test_long_wc_compliance_spmm	user defined bgl_subframe_rear_force
test_lat_par_compliance_spmm	bgl_subframe_rear_velo
test_lat_opp_compliance_spmm	bgr_subframe_front_disp
test_par_aligning_spmm (<a	bgr_subframe_front_force
test_opp_aligning_spmm (<a	bgr_subframe_front_velo
test_spmm (<acar_shared	bgr_subframe_rear_disp



# EXERCISE

- **Perform Workshop 19, “SVC AND SPMM EVENTS IN ADAMS CAR”.**

