ASTOS® 9

User Manual

Version: 9.16.0

Meitnerstraße 8
70563 Stuttgart
Germany

Printed in Germany.

December 2020

Preface

The *ASTOS* Software Project has been funded by the European Space Research and Technology Center (ESA/ESTEC), Noordwijk, Netherlands, under Contract No. 10136/92/NL/JG(SC) from 1989 - 2000. The contract monitor has been Klaus Mehlem.

Originally named *ALTOS* (Advanced Launcher Trajectory Optimization Software) the software evolved through various stages and with the participation of many individuals. In particular the following persons were directly involved in designing, implementing, testing and documenting the software:

Wolfgang Buhl, Klaus Ebert, Heinrich Herbst (Astrium GmbH, formerly DASA). Christian Jänsch, Zack Crues, Klaus Schnepper, Eugen Berger (Deutsches Zentrum für Luft- und Raumfahrt, DLR). Denis Fischer, Peter Gath, Alexander Hauer, Jörg Kindler, Albert Markl, Michael Paus, Axel Roenneke, and Klaus Well (Institut für Flugmechanik und Flugregelung, Universität Stuttgart, IFR).

In addition, several people contributed indirectly by providing software modules: Philip Gill (University of Southern California, USC), Kathy Horn, Dieter Kraft (DLR), Michael Rahn, Ullrich Schöttle (Institut für Raumfahrtsysteme, Universität Stuttgart, IRS).

The contributions of all team members are highly appreciated. Without their commitment and dedication, the goal could not have been achieved. Most of all, the continuous support and feedback from Klaus Mehlem at ESTEC in these years has given stability to the development team such that new software engineering concepts could be implemented successfully.

Since the summer of 2000 *ALTOS* has been renamed to *ASTOS* (Aero- Space Trajectory Optimization Software) due to a trademark conflict. Until 2006, *ASTOS* has been a commercial software package sold by the TTI GmbH, Department of Optimization, Guidance & Control.

Since October 2006 the former TTI department "ASTOS Solutions (Optimization, Guidance & Control)" has been spun-off, such that now development, sales and support for *ASTOS* have been taken over by Astos Solutions GmbH.

Since 2015 - in line with the extended functionality of the software - the meaning of the *ASTOS* acronym has been changed to "Analysis, Simulation and Trajectory Optimization Software for Space Scenarios".

PREFACE

Table of Contents

List of Figures

List of Tables

1 Introduction

ASTOS is designed for modelling a wide range of space scenarios (launch and re-entry, communication, navigation, Earth observation, science and exploration missions) and can be utilized for design, simulation, optimization and analysis purposes.

This user manual contains all essential information to make full use of the software. It provides an overview of features and capabilities and how these are integrated in a typical workflow:

- Chapter 2 introduces the graphical user interface of the software with all its windows and menus. It also provides a description of the available tools, general software settings and the various data input options.
- Chapter 3 gives an overview of the typical workflow with the software and of the various components which can be present in an *ASTOS* scenario.
- Chapter 4 covers all modules which are required to define a scenario. It provides a description of environmental models, how vehicles and other objects are set up from component templates and how their dynamics are specified.
- Chapter 5 introduces the available analysis features which can be applied to simulation data.
- Chapter 6 describes the workflow for an optimization of scenario parameters and gives an introduction of the corresponding settings and functionalities for this purpose.
- Chapter 7 gives a detailed description of the *Multibody dynamics* feature and of the required work-flow to perform a multibody simulation.
- Chapter 8 is dedicated to the inspection and presentation of results obtained with *ASTOS*. This covers the description and setup of plots and animations as well as the creation of reports and data exports.
- Chapter 9 introduces the built-in batch-mode capabilities of *ASTOS* and their use.
- Chapter 10 describes how *ASTOS* can be integrated in a development chain with other external tools. Supported export formats and required input formats are highlighted together with instructions on how to establish connections to databases.
- Chapter 11 summarizes abbreviations and common terminology used throughout the manual.
- Chapter 12 gives good practice guidelines and how to obtain support from Astos Solutions.

Note: This manual is not intended to cover all configuration details of every available model and its theoretical and mathematical background. The comprehensive description of models is given in the Model Reference.

2 User Interface Overview

This chapter shall give an overview on the graphical user interface of *ASTOS*. It provides a description of all relevant windows of the GUI in dedicated sub-chapters. Furthermore it gives information on how input data can be inserted into *ASTOS*, emphasising the non-trivial ASTOS-specific aspects. This chapter neither describes the logical relationship between the GUI components nor details on scenario settings. It shall rather give an overview on the GUI components. This background information is mandatory to understand the explanations given in consecutive chapters.

2.1 Main Window

A typical *ASTOS* scenario may easily contain more than a 100 settings. The layout of the main *ASTOS* window reflects this and tries to give fast access to all these settings in a clear manner.

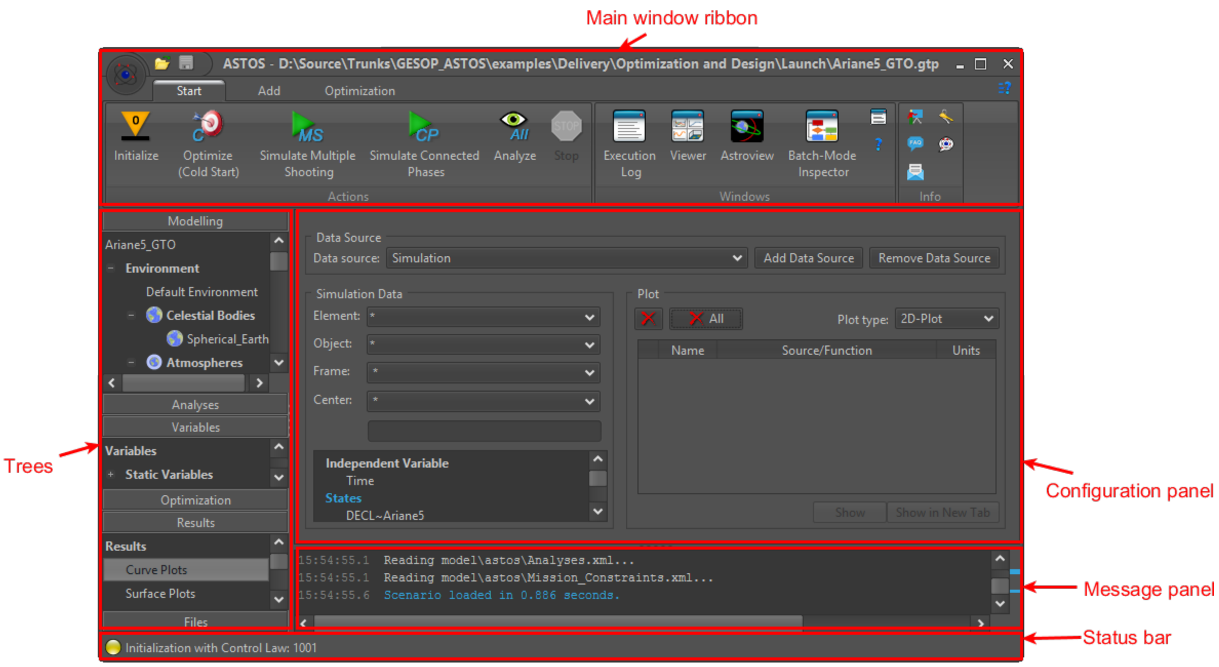

Fig. 2.1: Main window of ASTOS

The main window of *ASTOS* comprises a versatile ribbon, which is described in a dedicated sub-section. Below the ribbon, on the left, there are several trees. Each tree represents an aspect of the ASTOS scenario. The purpose of each tree is described in a dedicated chapter:

- *Modelling* - Chapter 4
- *Analyses* - Chapter 5
- *Variables* - Section 2.10.4
- *Optimization* - Chapter 6
- *Results* - Chapter 8
- *Files* - Chapter 3

The vertical space for a tree can be adjusted by dragging one of the tree title bars. A double-click on the title maximizes the corresponding tree view. Each tree node contains certain settings. When the node is selected using the mouse, these settings are shown in the **Configuration Panel** which is the panel on the right of the trees and below the Main window ribbon.

Below the Configuration panel, the Message Panel can be found. There time-tagged log entries are added by each action. Also some GUI-related error messages and warnings will be displayed in the Message Panel. Similar to the Execution Log (see Section 2.3) it provides an overview of the full log on the right of the vertical scroll bar, where warnings, errors and major entries are marked with corresponding colour bars.

Note: The Message Panel content is deleted when ASTOS is closed.

At the bottom of the main window a status bar indicates currently running actions or the result of the last action (lower left corner). A green ball is shown if the action was successful, a yellow ball is shown if the action was partially successful and in case of a red ball the last action failed. A sand glass indicates a currently running action. A message next to the ball gives more details about the status. Furthermore the status bar indicates unsaved modifications (showing a 3.5" floppy disk icon in the lower right corner).

2.1.1 Main Window Ribbon

As an alternative to menu and toolbar, ASTOS comprises a ribbon. A ribbon, also known from other modern software, allows fast and easy access to all actions. Its content depends on the currently activated and available features and also on the currently selected tree node, i.e. model panel content.

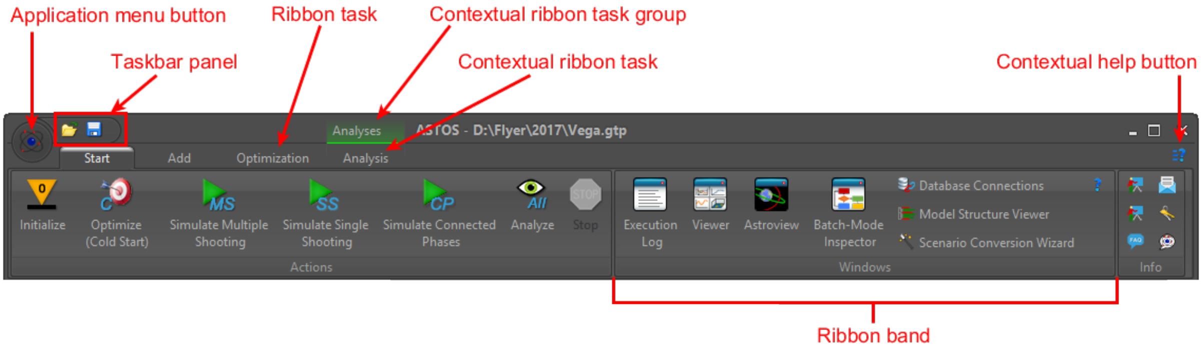

Fig. 2.2: Ribbon of the ASTOS main window

The *Application menu* comprises all scenario loading and saving options as well as Preferences and Quit button. It replaces a classical File menu.

The *Taskbar* panel (menu) holds frequently used actions. Since the taskbar is visible independent of the selected ribbon task, it serves as an alternative to the ribbon task action buttons. In future versions of ASTOS the content of the taskbar will be customizable.

Most of the other actions are organized in ribbon tasks, whereas the most frequently used are located in the *Start* task. Some tasks are so-called contextual tasks, i.e. they are only visible when a correlated entry in one of the trees is selected. Within a task, actions are grouped in ribbon bands. The action buttons are automatically resized depending on the available space in the *ASTOS* GUI, whereas first less important buttons are shrunken.

In the upper right corner of the ribbon, the *Contextual help* button is located. It opens a panel that displays the portion of the online help related to the currently displayed content of the *Configuration Panel*.

2.1.1.1 Application menu

The *Application menu* is accessible by clicking on the *ASTOS* icon in the upper left corner of the ribbon.

New Scenario
creates and opens a new scenario. Clicking on *New Scenario* causes the *New Scenario* window to appear. The following cells can be filled out:

- In the *Name of the scenario* field the desired name for the scenario shall be entered. A folder with this name and suffix `.gtp` will be automatically created to accommodate the scenario files and folders.

> **Note:** The suffix `.gtp` is automatically added to the scenario name.

- *Parent path* defines the path where the scenario (.gtp) folder from the previous step shall be stored
- The checkbox in front of the *Copy from existing scenario* field shall be checked and the path to the template scenario .gtp folder shall be specified in order to use an already available scenario as a template for the new scenario to be created. This is useful in case the user needs to create a working copy of an existing scenario, e.g. of an example scenario provided with the *ASTOS* installation. The example scenarios are available in the `%ASTOS%\examples\ASTOS\` folder (`%ASTOS%` is the installation path of the *ASTOS* software).
- The *Create scenario with wizard* checkbox shall be checked for a new scenario to be created and configured with the help of a scenario wizard. The wizard guides the user through a sequence of questions to configure the scenario step by step. The following scenario types are supported by the wizard:
  - *Ascent scenario (launcher, sounding rocket, amateur rocket) for trajectory optimization*
  - *Orbital scenario*
  - The *Multi-payload deployment* wizard supports the creation of a multi-payload deployment analysis scenario. The user needs to define step by step the following parameters:
    - central body
    - propagation type: 3-DoF or 6-DoF (Multibody simulation)
    - initial state (orbit) of the launcher (last stage)
    - center core element
    - number of dispensers and number of payloads
    - payload component (nominal orientation, size, mass, etc.)
    - number of deployment events and which payloads are jettisoned

    The generated scenario additionally provide the inter-satellite distance between the payloads and between each payload and the rocket (last stage) as output function.
  - *Other scenario (only a generic basic scenario)* create a basic scenario based on a central body and a vehicle type defined by the user.

Clicking on **Create** creates the scenario in case no *Create scenario with wizard* has been selected.

Open Scenario
Opens an existing scenario. A file dialog shows up where a `.gtp` folder or a `TOPS` file (if `.tops` is selected as file type) can be selected from the current folder. On the right part of the *Application menu*, a list of recently opened scenarios is available, when hovering the mouse pointer over the *Open Scenario* menu item.

Note: The provided example scenarios are read-only. To open them, remove the "read-only" flag or use the *Copy from existing scenario* option of the *New Scenario* action (see also Opening a scenario in the Getting Started manual).

All intermediate simulation or optimization results by any action in *ASTOS* are internally stored in a working `TOPS` file named `input.tops`. Select this file to continue working on a scenario or select any other `TOPS` file in the scenario folder to start from that result. Selecting any other `TOPS` file than `input.tops` displays a warning that the current working `TOPS` file will be overwritten.

Note: *ASTOS* loads the selected `TOPS` file, but all modifications to its data are again stored in the `input.tops`. To save a copy, use *Optimization→ TOPS→ Save TOPS As...*

Tip: On *Linux* platforms the opening of the `problem_intro.html` file with the default web browser requires the installation of the *Gnome* library. In the case of problems, please open the html file manually.

In case the selected scenario to be opened was created with an older version of *ASTOS*, a pop up window will appear giving the possibility that the scenario be converted to the current version. More details on the conversion process can be found in the Conversion chapter of the Migration from the Previous Major ASTOS Version book.

Reload Scenario
Reloads the current scenario such that all unsaved modifications will be reverted.

Save Scenario / Save Scenario As...
Saves the current scenario. The working `TOPS` is also saved each time a process (e.g. simulation or optimization) is executed. With *Save Scenario As...* it is possible to save the scenario to a different location. A complete copy of the working scenario is created, including any additional files present in the scenario folder.

Compress Scenario
Compresses the scenario folder into a zip-file and/or deletes unessential files created by *ASTOS* from the working folder. A dialog window appears where first the file categories can be selected which should be deleted. The individual categories are self-explanatory. Pressing the **Next** button shows all identified files and it is possible to deselect individual files to avoid their cancellation. On a final dialog page, it can be determined if a compressed (`.zip`) file which contains the complete scenario folder should be generated or the selected files should be simply deleted from the working folder.

Preferences
The *Preferences* window is shown as soon as the **Preferences** button is clicked. The button is located in the lower right corner of the *Application menu*. The content of the *Preferences* dialog is explained in chapter Section 2.9.

Quit

The **Quit** button is located in the lower right corner of the *Application menu*. It terminates the *ASTOS* application. If there are any unsaved modifications, a warning appears and the user is asked whether these changes should be saved before termination.

2.1.1.2 Start Task

Actions Band

The content of the Actions band strongly depends on the usage of the **Optimization** feature, i.e. in case the Optimization feature is installed and enabled, different buttons are available.

- *Initialize* generates an initial guess for the optimization. This button is only present if the **Optimization** feature is and enabled.
- *Simulate (Multiple Shooting/Single Shooting/Connected Phases)* starts a simulation of the scenario. In case the **Optimization** feature is available, the single *Simulate* button turns into three buttons. All options - namely multiple shooting, single shooting and connected phases simulation - are described in Section 6.5.
- *Optimize (Cold Start)* starts an optimization of the scenario. Only the commonly used cold start flavor is available in the Start task. The warm start variant is only accessible via the *Optimization* task. This button is only present if the **Optimization** feature is available and enabled (see Chapter 6 for details on the optimization topic).
- *Analyze* starts all analyses of the scenario in the user-defined order, no matter if their execution is set as automatic (see Chapter 5 for details on configuring and performing analyses).
- *Stop* stops a process running in the background, e.g. a simulation or an optimization. In case it is an optimization, it is possible to choose between either killing the current process without saving any intermediate result, or stopping the running process after the next major iteration step. The latter option saves the results obtained so far.
- *Save TOPS As...* saves the current state of a TOPS file. The saved file can be opened at a later time to restore the TOPS file. The complete description of the *Save TOPS As...* functionality is provided in the documentation of the .

Windows Band

The *Windows* band provides access to all important configuration and inspection windows that are part of the *ASTOS* GUI:

- The *Execution Log* window (Section 2.3) is used to display all output generated by an initialization, simulation, optimization or by any other *ASTOS* feature.
- The *Viewer* window (Section 2.2) contains all plots created for the current scenario.
- The *Astroview* (Section 2.4) opens the animation window of *ASTOS* in which the entire scenario is visualized using photo-realistic 3D computer animations.
- The *Batch-Mode Inspector* (Section 2.5) is used to setup and control batch mode sessions.

- Via the *Database Connections* window user credentials for database access and database connections are configured (see Section 10.3) .
- *Model Structure* option provides an overview on available states, parameters and constraints for each phase (after initialization/simulation of the scenario).
- *Scenario Conversion Wizard* opens a wizard that supports the guided conversion of the current scenario towards a different application.
- *ASTOS Help* opens the *ASTOS* online *Help* window that comprises manuals and tutorials (see Section 2.8).

Info Band

The actions in the *Info* band provide further general information and access to the support and license management features:

- *Scenario Introduction* displays a user-defined online documentation of the current scenario which is read from the `scenario_intro.html` located in the scenario directory. This file can be edited for online documentation or to provide instructions for other users.

> **Tip:** The scenario introduction can also be displayed automatically each time a scenario is loaded by selecting *Show introduction when opening a scenario* from the general settings in the *Preferences* menu.

> **Note:** On *Linux* platforms, opening the `scenario_intro.html` with the default web browser requires the installation of the *Gnome* library. In the case of problems, please open the `html` file manually.

- *Scenario Summary* creates and displays a HTML file containing a fixed report of the scenario containing the information on the scenario cost function, vehicle's initial position and model and on the environment models. For details, see Section 8.4.1.
- *Visit FAQ Website* opens the default Internet browser and visits the *ASTOS* FAQ (https://www.astos.de/faq) with helpful information on typical issues and questions.
- *Send Feedback* opens a new email in the default email client. Address and subject fields are already prepared and the message body already contains information about the *ASTOS* release and built number.
- *License Manager* provides a tabular overview on the license status and can also be used to activate or deactivate stand-alone licenses (see Section 2.6 for details).
- *About ASTOS* displays information about the *ASTOS* software including release and built number.

2.1.1.3 Add Task

The *Add* ribbon task contains various options which are required to set up the scenario From this menu, all kind of objects and templates (environment models, vehicles and components, analyses, etc.) can be added to the scenario. These options are grouped in the follwoing ribbon task bands:

- *Environment Models Band*
- *Vehicle Models*
- *Vehicles & Other Entities*
- *Dynamics*
- *Import*
- *Post-Processing*
- *Omptimization*

Import Band

The *Import* band offers the user the possibility to import any objects into a scenario from the following external resources:

- *From Database...* opens a window containing a list of pre-configured *ASTOS* scenario models which can be re-used in case of need. Details on how this model database shall be used are given in Section 2.1.1.3.1.
- *From Satsearch...* opens a window with an interface to the *Satsearch* database, which can be browsed by a search mask and used to create ASTOS model components with data from existing products. Details on this interface and on its use are given in Section 2.1.1.3.2.

2.1.1.3.1 From Database

Introduction

The model database in *ASTOS* is accessible through the *From Database...* menu, which opens a window containing a list of pre-configured *ASTOS* scenario models. These models are stored in a database which is split into a *Built-In* part, which is provided with the *ASTOS* installation and in a *Custom* part which contains models, created and added to the database by the user. The model database is a convenient way to save any *ASTOS* models of interest and re-use at a later time.

Description and use

The model database windows consists of a list of models grouped in categories on the left hand side and of a model description panel on the right. At the bottom of the window are situated the *Import* and *Custom* buttons.

Adding a model to a scenario
The user can select any of the models present in the list on the left. The model description text, the model version and source location are shown in the panel on the right. The user can import the selected model into the current scenario by clicking the *Import* button.

Adding a model to a database
Any model in the *Environment* and the *Vehicle Parts & Properties* nodes in the *Modelling* tree can be added to the model database via a right-click on the model and by clicking *Add to database*.

Background

The models in the *Built-In* part are stored in the <ASTOS_installation_folder>\Model_Database\Model_Database.xml file, which is generally write-protected. Thus the models in the *Built-In* part cannot be removed from the list.

The models in the *Custom* part are stored in
<user_folder>\AppData\Roaming\Astos Solutions\ASTOS\<version>\Model_Database\Model_Database.xml
(on *Microsoft Windows*)
or in
<user_home>\Astos Solutions\ASTOS\<version>\Model_Database\Model_Database.xml
(on *Linux*),
to which the user has write-access. Thus the models in the *Custom* part can be removed (right-click on the item in the *Custom* list and click *Delete item*)from the list.

2.1.1.3.2 From Satsearch

Introduction

Satsearch [44] is a company, which provides a database for space components produced by several hundred different companies from all over the world. Its aim is to build the first global marketplace for space and thereby to make the process of finding components required for a space engineering project easier and more efficient through presenting the product data centrally on their website and via an application programming interface (API). So, the process of searching manually for companies producing the desired equipment and browsing through their websites is shortened. In order to accelerate this process and especially trade-off studies with components from different suppliers even further, ASTOS provides an interface to the Satsearch-API which makes it possible to search for products with certain features and to import them with their properties directly into your ASTOS scenario as vehicle model.

Access the Satsearch database

In order to use this feature, it is required to sign up on the *Satsearch* website and to request user credentials for their API via mail. Thereafter the credentials (an *API Token*) have to be saved in the *Satsearch credentials* pane of the *Start*→ *Database Connections* window in the *ASTOS* GUI.

Usage of the ASTOS interface to Satsearch

The user can access the *Satsearch* interface via *Add*→ *Import*→ *From Satsearch...* in the *ASTOS* GUI. This opens the *Satsearch* window main view.

- ***Satsearch* window main view**
  The Satsearch main view contains the following panels:

  - In the *Search* panel the user can select the component *Category* and *Sub-category* in which the search shall take place by choosing from a drop-down menu. The shown categories are provided by the *Satsearch* database and not all of them contain any products yet. After the selection is finished, the search can be started by clicking the **Start search** button. The search can take a few moments and during this time the *Searching...* indicator is shown next to the greyed out **Start search** button.
  - The *Filters* panel contains options for filtering the search results:

    - Via the **Add filter** button a filter can be added to the *Active filters* panel.
    - With the **Update filters** button the selected filters are applied to the results. Only filters for which a filter value is provided by the user are applied. Multiple filters are applied using the AND operator. Filters whose values are not yet provided result in the filter quantity and its value being shown in the *Results* table.
    - The **Delete filter** button removes the last added filter.
    - In the *Active filters* sub panel the active filters are listed. The property, which shall be used as filter, can be selected via the *Property* drop-down menu. In order to show only products within a certain properties range (e.g. with a mass smaller or equal than 20 kg), the required relational operator (e.g. #) has to be selected in the *Operator* drop-down menu, the value (e.g. 20) has to be entered into the *Value* field after it and the corresponding unit (e.g. kg) has to be selected in the *Unit* drop-down menu of the filter row. Clicking the **Update filters** button applies the filters to the search results. At least one filter row is always present in the filter list. Any filter can stay unconfigured if needed.

  - In the *Results* panel a list of the products matching the selected categories and active filters are shown. By double-clicking an item int the list or by selecting an item and clicking the **Show selected product** button the product view is shown in the same window.

- **Product view and import**
  The product view shows all data which is provided by the *Satsearch* server about product chosen in the previous step. Besides it gives the opportunity to directly create an ASTOS component based on the found product without having to copy-paste the data manually. The product view contains the following areas:

  - The *Meta data* panel shows non-engineering data, such as the product name, supplier and the respective *Satsearch* URLs. Additionally a product image is shown if available.
  - The *Description* panel gives a short description of the product, which in some cases is needed for a better interpretation of the product data.
  - In the *Product attributes* panel the available engineering data of the product are listed as a table. The properties names in this panel are the ones used by the *Satsearch* data. Sometimes this data contains different versions and models of the component or different values for the same component property (e.g. mass), which are explained as a

comment. It is advisable to check which properties and values are adopted in the *Import data* panel.

- In the *Import data* panel the user can configure the component import into ASTOS as a vehicle model with the help of several drop-down menus. The *Vehicle Model*, *Type*, *Subtype* and the *Identifier* of the ASTOS model object are created automatically. Regarding the *Identifier* the usual ASTOS name rules have to be followed. The remaining settings in the *Import data* panel contain properties of the chosen ASTOS vehicle model which are provided by the *Satsearch* databese. In case of missing or incorrect data, the user can change the corresponding values later in the *ASTOS* vehicle part view.

Note: The *Vehicle Model*, *Type*, *Subtype* and the *Identifier* as well as the physical component properties in the *Import data* panel are populated automatically using the data in the *Product attributes* panel. It is not always possible that this process is error-free and the user should check if the values in the *Import data* panel are correct.

After the selection process is finished, the ASTOS model can be created by clicking the **Create Object** button. The ASTOS model part is immediately created and shown in the *ASTOS* GUI. By clicking on the **Back** button the user can return to the main view of the *Satsearch* interface.

- **Satsearch workflow**
  The suggested workflow when using the *From Satsearch...* import is as follows:

  - Click the **From Satsearch...** button to open the *Satsearch* database search window
  - Select component category and subcategory of interest
  - Click **Start search** button
  - Check the results
  - Optional:
    - Add filters
    - Select filter properties from drop-down menu and specify them
    - Click **Update filters** button
    - Check filtered results
  - Select an item in the *Results* table by double-clicking or selecting and clicking the **Show selected product** button in order to open the detailed product view
  - Check if product matches your requirements
  - Check if the pre-selected *ASTOS* model type and subtype fits the desired objects and modify them if needed
  - Check the identifier for being a valid *ASTOS* identifier
  - Check and select the best fitting properties values. If no value is present or fitting, leave the field empty and enter a value later in the *ASTOS* model
  - Click the **Create Object** button and if the component creation is successful, a corresponding dialog opens

- Return to the main *Satsearch* interface main view by clicking the **Back** button to continue the search or close the *Satsearch* window

2.1.1.4 Optimization Task

The *Optimization* task is only present if the **Optimization** feature is installed and the Optimization feature is enabled. It contains various actions which are required for optimizing the scenario results (refer to Chapter 6 for more details):

Actions Band

- *Initialize* initializes the scenario after all models have been defined. The initialization process creates the required grids and parameters.
- *Optimize (Cold Start/Warm Start)* starts the optimization process either using a cold start or a warm start approach. As a general rule, *Cold Start* serves as default action, in particular after a new grid has been generated or when a grid has been modified after the previous optimization run (e.g. when a new constraint has been activated). *Warm Start* option should be used only when a previous optimization did not converge within the specified maximum number of iterations.

> **Tip:** The *Cold Start* method normally results in a better optimization performance and should be used as default.

> **Tip:** It is recommended to run a simulation before optimizing the scenario. Doing so, the initial trajectories can be inspected before running an optimization.

- *Stop* stops a process running in the background, e.g. a simulation or an optimization. In case it is an optimization, it is possible to choose between either killing the current process without saving any intermediate result, or stopping the running process after the next major iteration step. The latter option saves the results obtained so far. This action is identical to the *Stop* button in the *Start* task.
- *Update States Bounds* restores the states bounds to values defined by the *ASTOS* model. Since these bounds might depend on the state values, the resulting bound values are not necessarily identical to those obtained after an initialization of the scenario, so this behaviour can be useful e.g. if the state bounds were to tight. Moreover, when a new major grid node is manually added, the bounds are linearly interpolated between already existing nodes, which might result in a "bad" placement of the resulting bounds. In this situation, *Update States Bounds* can be also beneficial.

TOPS Band

This ribbon band comprises options related to `TOPS` files. These files contain all necessary information to run an optimization or simulation of a scenario, e.g. optimization settings, phase information or grid data. The following actions are available:

- *Save TOPS As...* stores the working status of a `TOPS` file that can be reloaded afterwards by providing a filename and path. It is possible to save the file either in a binary format (`*.tops`) or in a `XML` format. The binary format is the one directly used by the software.

  Tip: This is the most convenient way to create a permanent `TOPS` back-up file.

  The `XML` version represents a human readable format of the `TOPS` file and is basically meant as a debugging aid for an expert user. It can be also loaded from the GUI using the *Open TOPS* option. The *Save TOPS As...* is present in both the *Start→Actions* and the *Start→Optimization* ribbon bands for convenience.

  Note: The `TOPS` does not contain any model data of the scenario. To explicitly save all model data, make a copy of the `model` subfolder of the working scenario.

- *New TOPS* creates a new, empty `TOPS` file. This `TOPS` file already contains all relevant data to describe the selected scenario (number of phases, number of states, controls etc.). However, it does not yet contain any grid data, so it cannot directly be used for an optimization or simulation run. With this basic `TOPS` data, it is possible to use other *ASTOS* tools to edit the setup of the optimization problem, especially the necessary grid data (see Section 6.4 and Section 6.7 for more details).

  Tip: There is no need to create a new `TOPS` file before an initialization. The *New TOPS* option is useful for debugging in case the initial guess generation is not successful (e.g. in case the equation of motions cannot be integrated due to singularities).

  Note: *New TOPS* resets all grid settings, optimizer settings, etc. in the current working `TOPS` file (`input.tops`) to default values!

  Note: The TOPS data is not directly written to file when pressing this button but just the data in memory is cleared. Therefore it is possible to recover the written data using the *Open TOPS* button.

- *Open TOPS* loads data from an existing `TOPS` file. This updates all settings which are not related to the scenario models itself to the values stored in the selected file.
- *Open Backup TOPS* loads data from an existing backup `TOPS` file (*.tops~). The backup `TOPS` `input.tops~` is always created when ASTOS wants to overwrite the `input.tops` file.
- *Update TOPS Description and Verify/Reinitialize TOPS* are mainly used for debugging purposes and are reserved for expert users in special situations. Generally, if there is a mismatch between the *ASTOS* model and the `TOPS` file, an automatic verification is executed.

2.1.1.5 Contextual Tasks

Contextual tasks are available as soon a correlated tree node is selected. These tasks are typically organized in contextual task groups but currently there is never more than one task in a contextual task group. The following contextual tasks are available:

- *Model* is available for equipment and environment models as well as for scenario entities (vehicles, ground stations, POIs, etc.). It gives access to *Rename*, *Clone* and *Delete* actions.
- *Phase* provides access to *Merge*, *Split*, *Rename*, *Clone* and *Delete* actions for phases of the Dynamics Configuration.
- *Analysis* comprises actions to perform analyses, to create their default report templates, to set the *Automatic* and *ASOR* flags, to change the sequence of the analyses and, similar to the *Model* task, actions to *Rename*, *Clone* and *Delete* an analysis. Details on the analysis topic can be found in Chapter 5.
- *Report* comprises actions to create reports defined in the scenario and actions to *Rename*, *Clone* and *Delete* a report. Details on how to work with the report functionality can be found in Section 8.4.
- *Export* comprises actions to perform exports previously defined in the scenario and actions to *Rename*, *Clone* and *Delete* an export (see also Section 10.2).

2.2 Viewer

The *Viewer* tool in *ASTOS* is used to accommodate the diagrams created as follows:

- from the *Curve Plots* (see Section 8.2.1) and *Condition Plots* (Section 8.2.3) nodes in the *Results* tree
- from the *Grids* node (Section 6.7) in the *Optimization* tree
- from the *Review Iterations* (Section 6.12) in the *Optimization* tree

The *Viewer* window is opened automatically each time a diagram has been created by the user. Additionally the following possibilities exits to open the *Viewer* window:

- using the **Show** or **Show in New Tab** buttons in the nodes mentioned above
- using the ribbon *Start*→*Windows*→*Viewer* action button

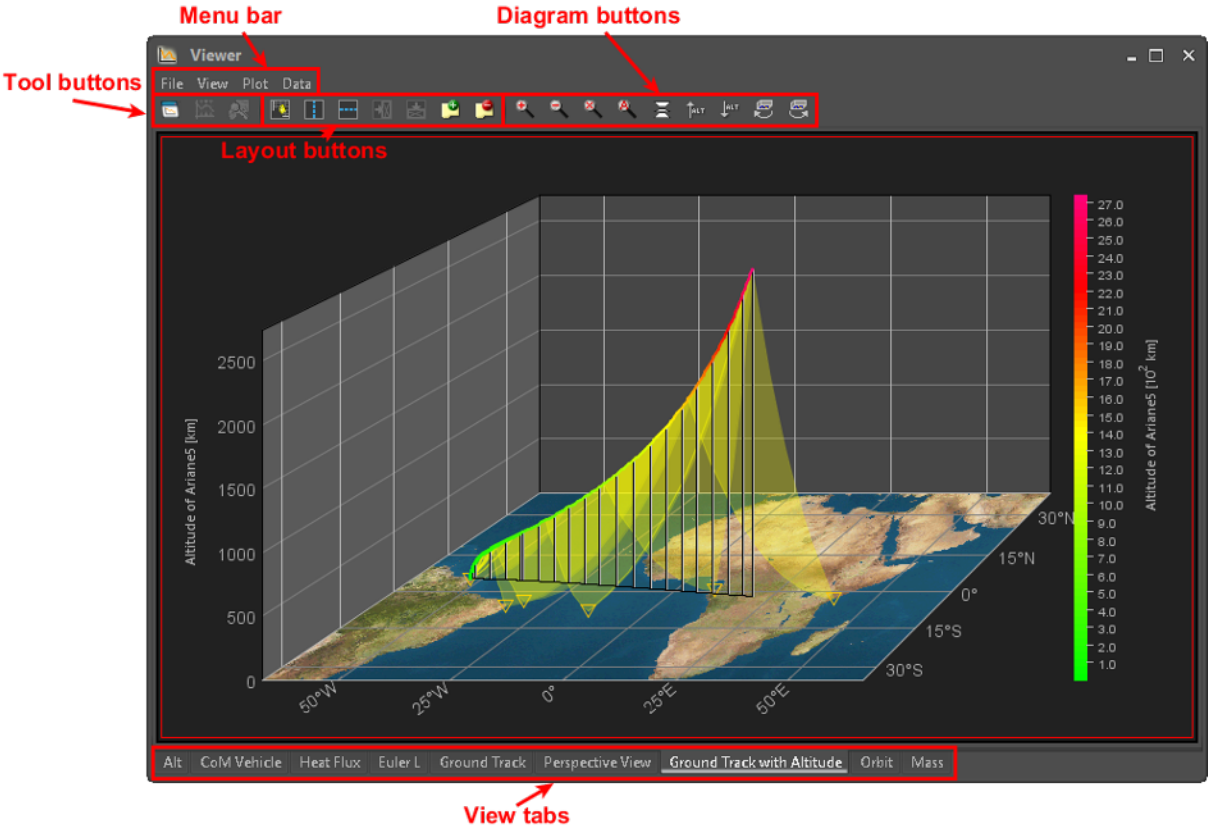

Fig. 2.3: The Viewer window

The *Viewer* window can accommodate multiple plots (**Views**) which are stored in dedicated tabs. The central area of the *Viewer* window shows the **View** whose tab is selected in the tabs selection area in the bottom part of the *Viewer* window. Tabs can be renamed by double-clicking on the tab and entering the new tab name. A view is either empty or displays one or several diagrams. The selected diagram features a red border.

There is a menu bar located at the top of the *Viewer* window. The number of available menus inside depends on the currently selected diagram type. The first two menus, (*File* and *View*), are always present and their content is identical for all kinds of diagrams. Other menus have options which depend on the diagram type (*Data*) or are only available for certain diagram types (*Plot* and *Axes*).

The toolbar is split into three sections. The **tool buttons** are common to all diagram types and provide shortcuts (mouse left-click) to the *Curve Plots*, the *Grid* and the *Review Iterations* configuration panels. The **layout buttons** are also common to all diagrams. They serve as shortcuts to all menu entries of the *View* menu, e.g. splitting or merging the view area. The **diagram buttons** depend on the diagram type. All commands available in this section are explained in a separate chapter in Section 8.2.

2.2.1 Viewer Menus

File Menu

The options of the *File* menu are related to commonly used commands:

- *Save as...* saves the window settings (selected items, layout, number of tabs etc.) as *ASTOS* plot files (`*.gavc`). It is important to remember that these settings are automatically saved (and loaded) in `default.gavc` any time a scenario is closed (or re-opened).
- *Load* opens an existing `.gavc` file. The content of the *Viewer* window is replaced by the content stored in the selected `.gavc` file.
- *Empty* removes all views from the *Viewer* window. Afterwards only a single empty view with the default name *View 1* is available. A new diagram can be created using the *Curve Plots* panel. Before removing all views, a short dialog opens which needs to be confirmed in order to remove all views.
- *Export Selected Plot As...* allows to save the selected diagram as an image in a commonly used format. Supported image formats are `.png`, `.bmp` and `.jpg`. Some basic functionalities are provided to set the desired dimension of the saved image.
- *Print Selected Plot* prints the selected diagram. The selected diagram is marked with a red border.
- *Print All* prints the whole view of the currently selected tab.
- *Print All Tabs* prints all the tabs available in the *Viewer* window.
- *Close* closes the *Viewer* window.

View Menu

The options of the *View* menu give the possibility to arrange several diagrams in multiple views or even to adjust up to 16 diagrams in one single view. After selecting a diagram with a mouse-click (selected diagrams are framed in red), the *View* menu offers the following options (alternatively, the corresponding button of the toolbar can be used):

- *Clear* clears the selected (red framed) diagram.

> **Tip:** If a *Curve Plots* diagram is currently displayed, it is necessary to clear the plot to reactivate the *Grids* and *Review Iterations* toolbar buttons (Fig. 2.3), .

- *Tile Vertically/Horizontally* splits the area with the selected diagram vertically or horizontally. To insert a diagram into a particular section of a view, first activate this section by clicking on it (red framed afterwards). Second, select the items to be plotted in the *Curve Plots*, *Grids* or *Review Iterations* configuration panel.

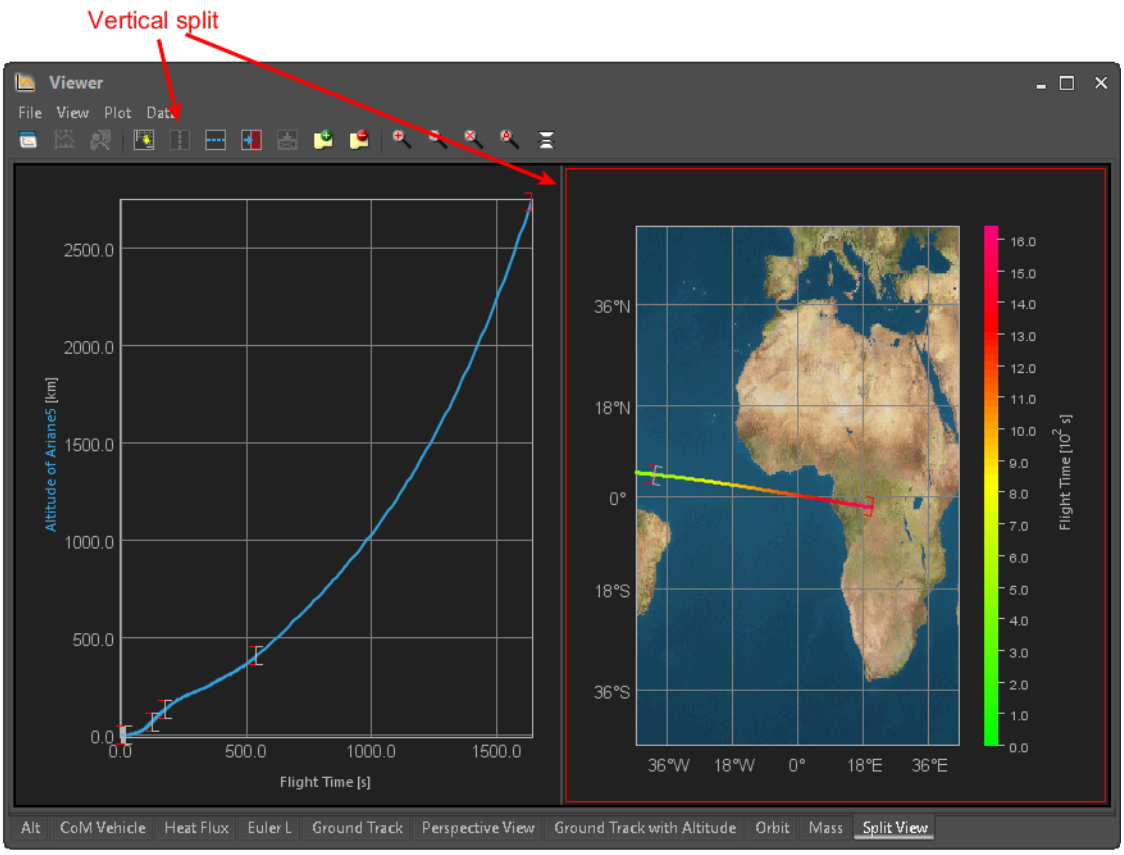

Fig. 2.4: The Viewer window can be split into sections (here: a vertical split)

- *Merge Vertically/Horizontally* reduces the number of tiles arranged vertically or horizontally by deleting the marked diagram.

> **Note:** The merge command always deletes the selected diagram (red framed).

- *New/Remove View* adds/removes a new view tab in the *Viewer* window. To be distinguishable, all the views are serially numbered. To switch between the views, select the desired view tab in the register (lower part of the *Viewer* window, see Fig. 2.3).

> **Tip:** The name of the view can be modified by double-clicking on the tab label.

Plot Menu

> **Note:** The following menus and menu options mainly depend on the underlying *Plot type*. For a detailed description of all types and their purpose, refer to Section 8.2.

The *Plot* menu contains general options to adapt the currently selected diagram. The entire menu is not available for a *3D-Plot*. After selection of a diagram which provides the *Plot* menu, the following options can be present:

- *Zoom In/Out* increases or decreases the displayed area in the selected diagram (not available for a *Satellite Plot*).
- *Fit Data* adjusts the displayed area in the selected diagram such that the plot data is shown as large as possible.
- *Show Legend* enables or disables a legend for the currently selected diagram. A legend is only displayed if there is relevant data to be distinguished.
- *Background...* opens a dialog which contains a list of available map backgrounds, e.g. visual Earth surface (not available for a *2D-Plot*).

Data Menu

The *Data* menu contains options to customize the line styles or to add additional visual entities to the diagram, e.g. events. This menu is always present, but the content slightly differs depending on the selected diagram. In general, the following options can be present:

- *Events* provides a popup menu which contains a list of predefined events, e.g. when the maximum heatflux or acceleration is reached along a trajectory (not available for a *3D plot*). Events can directly be selected (or deselected using) in the popup menu and are then displayed in the associated diagram. The *Events...* submenu opens a dialog where the visual representation (symbols, colour, labels) for each event can be specified individually.
- *Line Style...* opens a dialog where the line style for plotted data can be selected (not available for a *3D-Plot*). It is possible to configure the colour, line stroke and line width for each plotted data line individually. By default, the data source representation is determined by the line stroke and the data itself by the line colour. The option *Use alternative line style scheme* switches this assignment, i.e. colours are used to distinguish data sources and the line strokes are used to distinguish the data of one source.
- *Gradient Style...* opens a dialog where a gradient style can be defined for data defined as a *Line Color* (only available for *Map-Plot*, *3D Map-Plot* and *Satellite Plot*). The gradient style is used to visualize additional data on a data line as colour gradient (e.g. heatflux data on a plotted trajectory). The dialog provides several options

  The *Current / Collected Example(s)* panel is used to choose an existing gradient style or to define a new style with specified *Name* to produce a sort of "style gallery". Gradient styles can also be imported or exported with the respective buttons which load or save files of a specific format (`*.gagc`).

  The *Gradient* panel is used to configure a selected gradient style. A gradient is defined by various *Stops* of a certain *Color* which can be chosen from a palette. The *Location* of a stop on the colour bar can modified by providing its relative position in percent. Alternatively, a stop mark below the colour bar can directly be dragged to a new position with the mouse. New stops can be introduced with the **Add** button, existing ones can be deleted with the **Remove** respectively when selected.

  Smoothness determines if only "discrete" colours or a real colour gradient is displayed.
- *Auxiliary Items...* is only available in case auxiliary items (see *Curve Plots*) are defined in the data source. Otherwise this option is greyed out. This option opens a dialog where auxiliary items to be displayed can be selected or deselected. Additionally, the visual representation for each item (symbol, colour, labels) can be defined.

- *Ground Stations ...* is used to enable or disable the visualization of ground stations (only available in the *Map-Plot* and in the *3D Map-Plot*). This option opens a dialog, where ground stations can be selected from a predefined catalogue of existing ground stations. Additionally, the visual representation for each ground station and its corresponding field of view can be specified (symbol, colour, labels and opacity of the field of view to display visibility regions). The *Preview* panel also provides information about the ground station (identifier and coordinates). In a *Map-Plot* the *Reference orbit altitude* has to be specified in order to plot the ground station visibility circle at this altitude. Fig. 2.5 shows a *Map-Plot* with all built-in ground stations available in *ASTOS* and their corresponding visibility circles at 10000.0 meters of altitude.

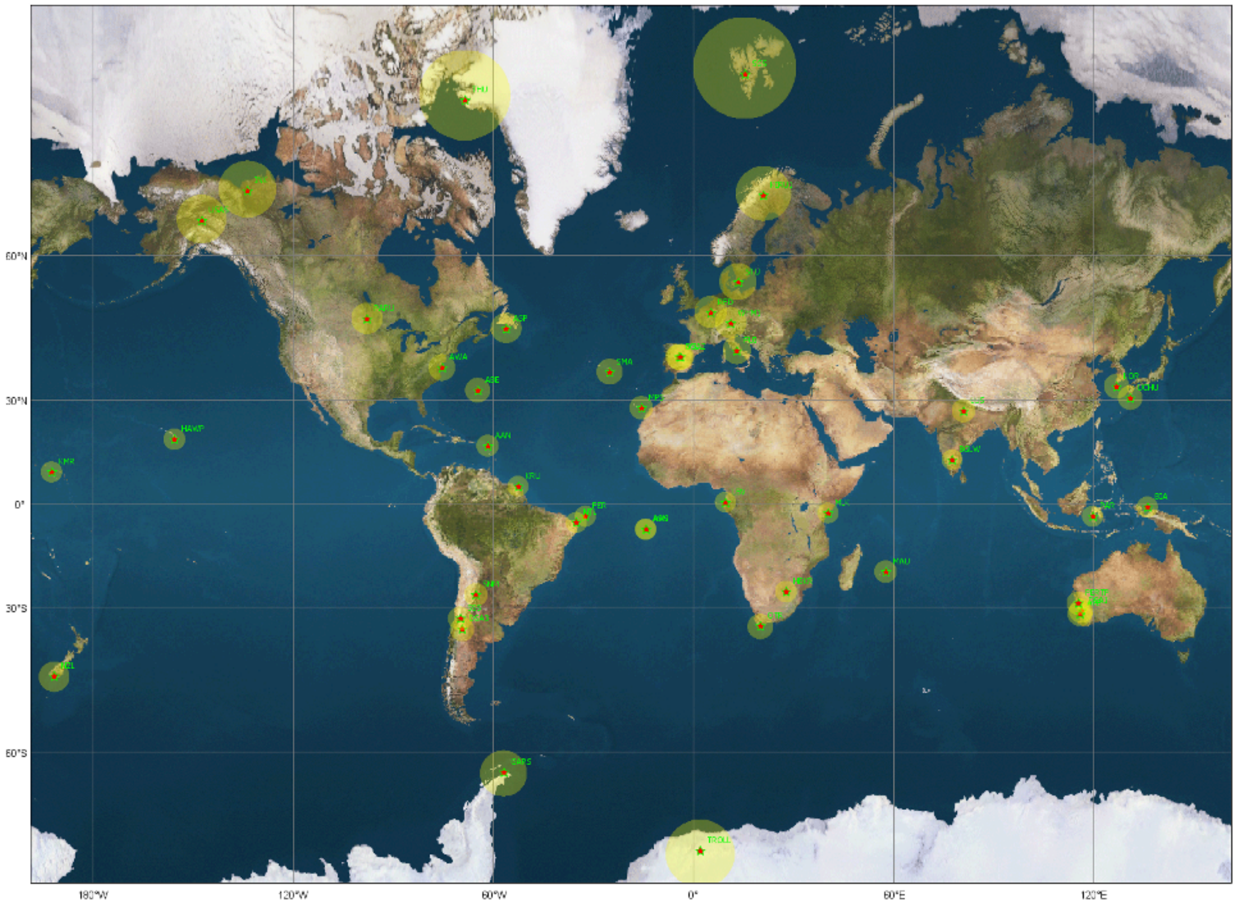

Fig. 2.5: Built-in ground stations in ASTOS. ASTOS

Note: The field of view and the visibility circles only show the trajectory visibility and are not considered as constraints. In order to define a visibility constraint, refer to Position Related Constraints in the Model Reference.

- *Show Phase Borders* can be used to highlight transitions between phases (not available for a *3D-Plot*). If enabled, a square bracket is drawn on the data line at each end of a phase.

Axes Menu

The *Axes* menu contains all options necessary to customize the appearance of the diagram axes. Apart from *Fixed Aspect Ratio* (for a *3D-Plot*), all following options are only available for a *2D-Plot*:

- *X* opens a popup menu where abscissa scale can be switched from linear to *Logarithmic scale* and vice versa. Additionally, the axis unit (see Section 2.10.3.1) can be specified here.
- *Y1/Y2/Y3/Y4* opens a popup menu in which the corresponding ordinate scale can be switched from linear to *Logarithmic scale* and vice versa. Additionally, the ordinate units (see Section 2.10.3.1) can be specified here. The *2D-Plot* supports up to four different y-axis. In case less axes are available in the plot, the respective options are greyed out.
- *Fixed Aspect Ratio* is only available for a *2D-Plot* in case one y-axis is defined and the units of the x- and y-axis are compatible, but always for a *3D-Plot*. Selecting this option results in a common resolution for the x- and y-axis. Normally, the resolutions for the axes can differ.
- *Use Colored Captions* is used to colourize the axis labels of a *2D-Plot*. By default, the axis label colour matches the colour of the related data line. When disabled, all axis labels are displayed in the default text color of the *ASTOS* GUI.

2.3 Execution Log

The *Execution Log* window displays all textual output created by the various *ASTOS* actions, e.g. simulations or optimizations. All messages created by these actions appear in order of their appearance. Normal output appears in grey whereas status messages appears in blue, warnings in yellow and error messages in red. New messages are appended at the end and a lateral scroll bar allows to display older messages and colour identifiers (e.g. blue markers for status messages). The content of the *Execution Log* can be copied with the shortcut *Ctrl+C*. Additionally the following buttons are present:

-  Clears all the messages present in the *Execution Log*. Note that there is no "Undo" for this operation.
-  Saves the current content of the *Execution Log* to a text file, e.g. to keep a protocol of an optimization run. A *Save Log* window appears where a location and file name can be selected.
-  Prevents the text from being scrolled automatically to the end of the list, e.g. to read a message while other messages are coming in. To deactivate the hold function, press the button again.

2.4 Astroview

Astroview is the name of the ASTOS built-in animation tool. With *Astroview* the currently loaded space scenario and its simulation results can be realistically displayed, optionally overlaid with visual representations for mission analysis results. The user can control the speed of the animation and can scroll through the simulation time. Viewpoint and field of view can be customized. Visual aids overlaid onto the scenario can be switched and customized in terms of colour and transparency. The animations can be recorded into standard video formats, supporting user defined key frames for animation speed and viewpoint.

Information on how to use *Astroview* is given in Section 8.3.

2.5 Batch-Mode Inspector

The *Batch-Mode Inspector* is one option to create and run batch processes in *ASTOS*.
It uses a dedicated GUI window to configure all batch parameters in a script. The batch structure is visualized as a tree whose nodes represent the various process commands/ jobs of the batch process. The *Batch-Mode Inspector* can be accessed using the *Start→Windows→Batch-Mode Inspector* action button . All batch mode commands and functions are explained in detail in Section 9.1.

2.6 License Manager

ASTOS grants license rights for features in two different ways. One way is to store license rights locally in the *Trusted Storage* which provides the possibility to activate or deactivate licenses directly in *ASTOS* using an Internet connection (or offline, if no Internet connection is available). Such locally stored licenses are called node locked licenses.
Alternatively, license rights can also be stored on a license server and then be shared by several computers. This type of license is called server or floating license.

Checking the status of available license features as well as license activation, deactivation and repair is all managed using the *License Manager*. It can be accessed using the *Start→Info→License Manager* action button in the *ASTOS* GUI. The *License Manager* window basically contains two tabs which are further described below.

Features

The first tab (see Fig. 2.6) shows the status of available licenses and features.

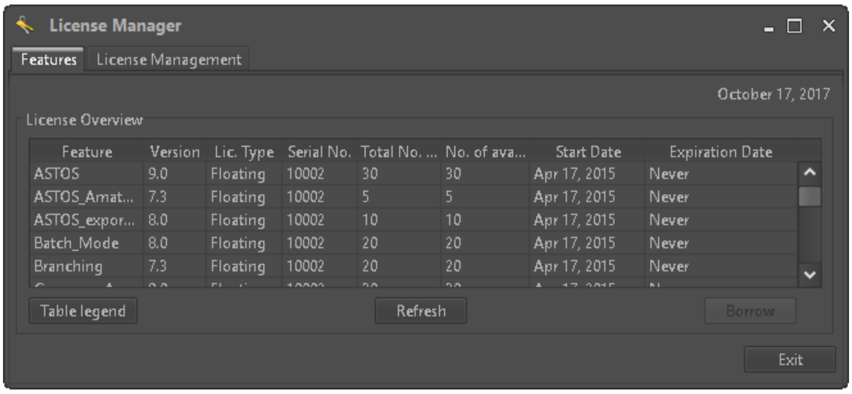

Fig. 2.6: The Features tab

All the features (1st column in the *License Overview* table) are listed with the specific expiration date (last column) if applicable. The number of available licenses (5th and 6th column) is relevant only in case of a server license. In case of a node locked license, *n/a* (not applicable) is displayed. For the status of different features, a certain colour-coding is used:

- Grey color identifies valid features.
- Yellow is used for features that will expire soon.
- Red colour highlights features that have already expired or are broken.

Note: In case a license feature is stored in the Trusted Storage and is marked as broken, a license repair is required. Please contact service@astos.de in case of issues.

License management tab

The second tab (see Fig. 2.7) is used for all kinds of license management activities. Here, a full list of available fulfilment records is shown. Each fulfilment record contains a set of license features and an expiration date.

Note: In case only floating licenses are available for all features, the *Fulfillment Record Overview* is empty.

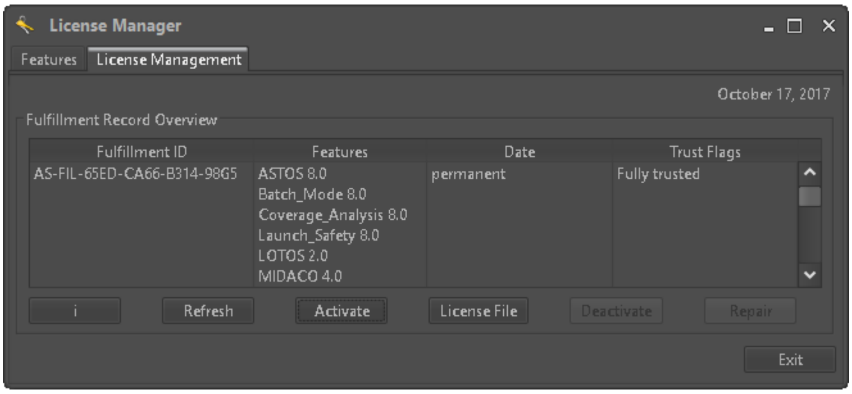

Fig. 2.7: The License Management tab

For each fulfilment record, the following information is provided:

- A unique *Fulfillment ID* which describes the set of activated license features.
- The list of license *Features* stored in the fulfilment record.
- The expiration *Date* of the fulfilment record and the associated license features.
- The *Trust Flags* of the fulfilment record. Possible values are *Fully trusted* and *Broken*.

Below the overview table, several buttons are available. The **i** button opens the *Table Legend* with a brief explanation of the colour-coding used in the table. The **Refresh** button updates the license information shown in the *License Manager*. The remaining buttons are used to manage licenses and are explained in the following chapters. **Deactivate** is only accessible in case one of the fulfilment records is selected. **Repair** is only accessible in case a broken fulfilment record is selected.

2.6.1 License Activation

The activation of a license is initialized by a click on the **Activate** button in the *License Management* tab. This opens the *License Activation* dialog which guides the user through the activation process. The first page of the dialog is shown below.

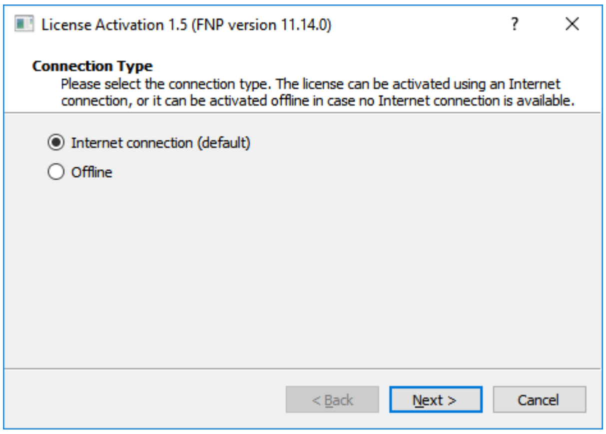

Fig. 2.8: Activation mode selection

Here, the connection type needs to be selected. In case an Internet connection is available, *Internet connection* should be selected. This is the preferred way for activating, deactivating or repairing a license. In case no Internet connection is available, *Offline* needs to be selected. Offline activation is explained in detail in Section 2.6.3.

On the next page (see Fig. 2.9), the *Entitlement ID* for the license is required; this can be retrieved from the web portal. To confirm the license activation, it needs to be provided the *Access code* that has been received via email and that is also used to access the web portal.

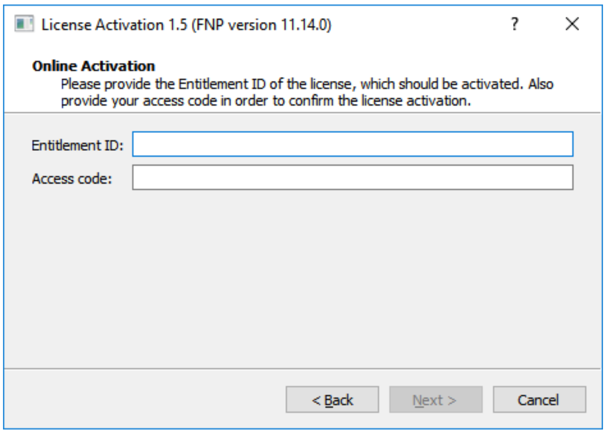

Fig. 2.9: Required license information for the online activation mode

A click on the **Next** button starts the activation process. A new page opens that shows the result of the license activation. In case the activation fails, the **Back** button can be used to check the provided data for the activation.

2.6.2 License Deactivation and Repair

To deactivate or repair a license, first a fulfilment record needs to be selected from the fulfilment record table. This enables the **Deactivate** button. In case the selected fulfilment record is broken, the **Repair** button is enabled.

Note: A broken fulfilment record needs to be repaired before it can be deactivated.

Tip: Deactivation of a node locked license is necessary for instance, when moving an *ASTOS* installation from one computer to another.

A left-click on the **Deactivate** button opens the *License Deactivation* dialog shown below. The *Fulfillment ID* is already selected. To confirm the deactivation, the *Access code* (received via email), which is also used for the web portal, must be provided. In case of a license repair, a similar dialog page is present.

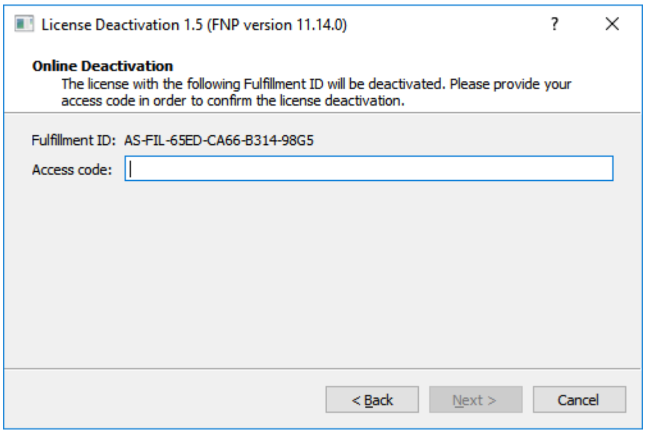

Fig. 2.10: Required license information for the online deactivation mode

A click on the **Next** button starts the deactivation or repair process. A new page opens that shows the result of the action. In case the action fails, the **Back** button can be used to check the provided data.

2.6.3 Offline Activation

Licenses can alternatively be activated offline in case no Internet connection is available. The offline license activation is started by clicking on the **Activate** button in the *License Management* tab. In the *License Activation* dialog, *Offline* needs to be selected before proceeding with the **Next** button. This opens the dialog shown in Fig. 2.11 where the *Entitlement ID* and the *Access code* are required. This step does not differ from the online activation.

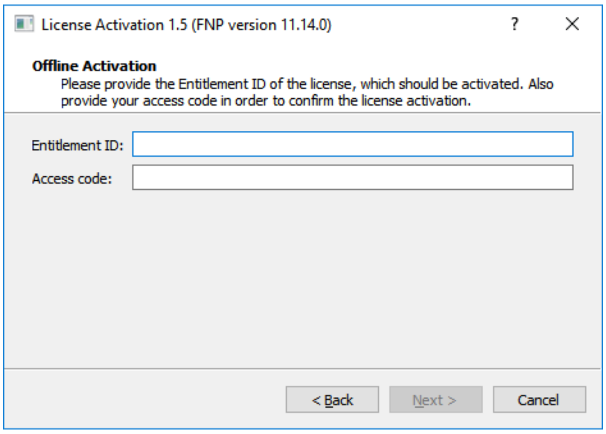

Fig. 2.11: Required license information for the offline activation mode

A click on the **Next** button opens the next dialog page (Fig. 2.12). Here, the activation request is shown. The content of this request needs to be copied to a text file.

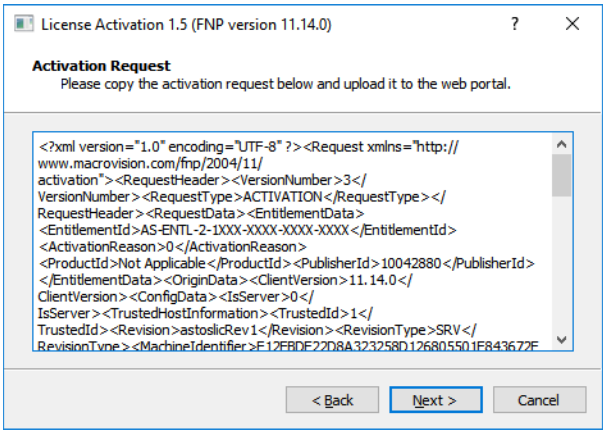

Fig. 2.12: The offline activation request needs to be copied from this page

This text file is necessary in order to generate a response file on the web portal. The generated response file then needs to be loaded on the next dialog page to proceed with the license activation. This page is shown below.

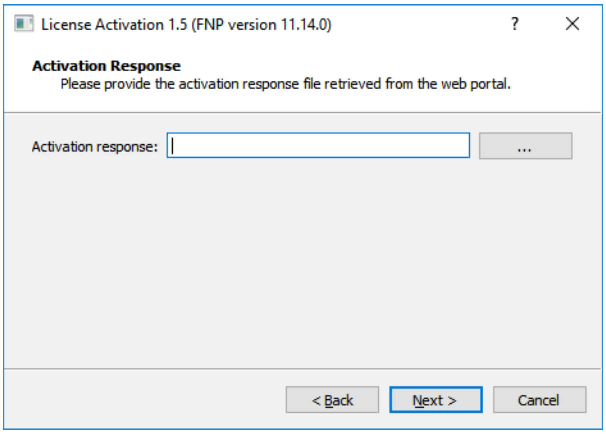

Fig. 2.13: The file containing the offline activation response needs to be specified on this page

The **Next** button starts the activation process. A new dialog page is open to show the result of the license activation. In case the activation fails, the **Back** button can be used to check the provided data for the activation. The offline license deactivation or repair is performed in the same way using similar pages.

2.7 Verify Manager

ASTOS stores scenario information in two locations: grid data are saved in the `TOPS` file while the model is saved in `XML` files (in `\model\astos`). Some information is saved to both locations, e.g. the number of phases, states, constraints and parameters.

During an initialization event, these data are synchronized: the information is read from the model and saved in the TOPS file. When starting a simulation or optimization event, the consistency of the two data sources are verified and in case that a discrepancy is detected a user interaction might be required to solve it.

Note: The Verify Manager is only relevant if the **Optimization** feature is available. Otherwise, a simulation is always preceded by a initialization and the affected files are synchronized.

The GUI provides dedicated windows to display the differences and to support the user in synchronizing the data.

2.7.1 Phase Verify Manager

In case the **number of phases differs** between the model and the TOPS file, the *Phase Verify Manager* automatically opens to inform about the discrepancy and to receive required actions. The window is separated in 3 columns: the list on the left contains phases present in the *TOPS* file that the user can select, the list on the right contains phases present in the *Model*, the central part of the window contains action buttons to act on the *TOPS* resolving the discrepancy:

- **Move Up**/**Move Down** the selected phase in a list.
- **Merge** two selected *TOPS* phases into one phase.
- **Split** the selected phase into two phases. A dialog opens to select the major grid node where the phase should be split. Splitting is only possible where a major grid node is present.

  Tip: Add a major grid node to a state using the Grid viewer (Section 6.7) before splitting a phase.

- **Insert** a new phase. A dialog opens to specify if the new phase should be added before or after the selected one. The new phase is initialized according to the settings specified in *Modelling* tree → *Dynamics Configuration* → *Vehicle & POIs Dynamics*.
- **Remove** the selected phase from a list.
- **Import** one or more phases from another TOPS file. A dialog opens to select the TOPS file to be used.

OK confirms the applied modifications.

2.7.2 Structure Mapping

In case there is a discrepancy in the number or names of constraints, parameters, states or controls between TOPS and model, the *Structure Mapping* window opens automatically. Usually, no user interaction is required and this window is more "informative" than "operative".

The tree on the left lists all present phases, in the right panel the content of the *TOPS* is listed on the left and the content of the *Model* on the right. Phases that contain discrepancies are highlighted with a warning mark ( ). Each phase node can be expanded to further locate and visualize the discrepancy: e.g. a new final boundary constraint is present in the model but not yet in the TOPS file.

In case of a name discrepancy (e.g. a component was renamed in the model), the corresponding real parameters present a different name. In this situation they should be linked by the user with the mouse: i.e. mouse left-click on the empty circle near the not connected real parameter in the *TOPS* list, second mouse left-click on the respective empty circle near the real parameter in the *Model* list. A dialog window opens to confirm that the created link should be updated in all the phases where these parameters are present. The procedure is the same for states, controls, constraints and cost functions.

OK confirms all applied modifications or simply lets the automatic adjustment feature create missing elements (constraints, parameter, controls) and required links.

Manual Check

Although a lot of effort has been placed in the automatic procedure of the verify manager, some situations still require the user to actively perform checks and updates. Hereafter there is a list of the most common situations that require a user check:

- When a new control is added, sometimes the upper and lower bounds are not correctly set, use the Grid viewer (Section 6.7) to check and update the control grids.
- When a new parameter is added (e.g. payload optimizable), sometimes the parameter value and the upper/lower bounds are not set in line to the model but to 0.0. The *Real Parameters* node of the *Optimization* tree should be used to check and update the values.
- If the automatic renaming is not correctly working (i.e. error after the verify manager), it is suggested to use the ribbon *Optimization*→ *TOPS*→ *Verify/Reinitialize TOPS* button and then perform a simulation.
- When the phase A is split in phase A1 and A2, the final time of phase A1 is set identical to its lower and upper limit. If the split time should be optimizable, manually open the lower/upper bounds in *Phase Overview* node of the *Optimization* tree (Section 6.10).

2.8 Online Help

ASTOS is provided with an extensive set of information on how to use the software. This information is accommodated in the *ASTOS* Help. The user can access the help via *Start*→ *Windows*→ *ASTOS Help* in the GUI. This opens the *ASTOS Help* which contains the online version of the help.

> **Tip:** During the *ASTOS* installation the user can choose that the PDF version of the help is installed. The latter is then placed in the `help_pdf` folder of the *ASTOS* installation. Each book of the help is accommodated in a separate `.pdf` file.

ASTOS Help window

The *ASTOS Help* window contains the following areas:

- A toolbar in the upper left corner where navigation buttons are placed
- The **Contents** tab containing the tree-like contents view of the help
- The **Search** tab where the user can search for a string in the complete help
- The *Find in Page* field in the upper right corner which allows the user to search content of the current page in the help. Also the *Ctrl+F* shortcut can be used to search the content of the current page in the help.
- The main window area where the content of one chapter at a time is shown. The tree node corresponding to a chapter is then highlighted in the *Contents* tab tree.

> **Tip:** Depending on the installed Internet browser, on *Microsoft Windows* the user can zoom in and out the content of the current page in the online help using the *Ctrl* + mouse wheel up/down shortcut.

The help is composed of several books. Please use the tree on the left to navigate through the following topics:

- **Release Notes** lists the innovations and backward compatibility issues of the *ASTOS* versions.
- **Installation Guide** provides necessary information to install the software.
- **Getting Started** is intended as the entry point for each new user, it also contains a fast tutorial.
- **Migration from the Previous Major *ASTOS* Version** provides users of the previous major version of the software with an explanation of how to convert a scenario to the current version.
- **User Manual** contains a detailed description of the software functionality and explains how to use *ASTOS*.
- **Model Reference** contains a detailed description of all the models available in the *ASTOS* library.
- **Optimization Theory and Description of Methods** contains some theoretical information about transcription and optimization, plus the description of the optimization settings.
- **Simulink Interface Manual** describes how to couple *ASTOS* with *MATLAB Simulink*, the use of the *ASTOS* Model Simulink block and the *ASTOS* tool box.
- **Tutorials** contains some interesting tutorials about typical *ASTOS* applications and advanced features.

Text formatting

The following list contains explanations of the text formatting used throughout the *ASTOS* help and its meaning.

- **Button Formatting** is used for buttons (not toggle buttons and not buttons of the toolbar if their icon is printed additionally). Example: **Initialize**, **Multiple Shooting**, **Save**, etc.
- `Inline Source Code Formatting` is used for source code examples within the help text. Examples: The function `Terminal_Cost` returns the cost value evaluated at the final time.
- ```
  A source code block is used e.g. for all function declarations with their parameters.
  In order to indicate line wrapping the ↲
  character is used.
  ```
- *List Item Formatting* is used for items in drop-down lists and toggle buttons, as well as for model types. Example: In the *Add actuator* window, *Type* list: *Rocket*, *Jet Engine* or *User* use this formatting.
- *Equation Formatting* is used for symbols for/in equations in the help text
- **Feature Formatting** is used for license features or features that can be activated in the root node of the *Modelling* tree. Example: The **Optimization** feature must be present in the license and enabled from the root node of the *Modelling* tree in the *ASTOS* GUI in order to be able to set up a scenario for this task.
- *Field Content Formatting* is used for the content entered by the user in a input field/cell/table, etc. Example: *6378* (input of Radius of a celestial body).
- `Path Formatting` is used for the path to a file or a file itself. Also for command line commands with their parameters. Examples:`%ASTOS%\examples`, `simulation.txt`.
- *Label Formatting* is used for all labels for input fields, selectors, buttons in the model configuration panel. Example: *Normalized time*, *Mission start date*, *Total mass*, etc.
- `Log Formatting` is used for the log output in the execution log and message panel. Example: `"Scenario loaded in 0.45 seconds."`
- *Menu Formatting* is used for all GUI menu items. Example: *Compress Scenario*.
- *Name Formatting* is used for all proper names or acronyms used as a name. Examples: *ASTOS*, *PROMIS*, *Google*, etc.
- *Node Formatting* is used for all tree nodes in the scenario modelling trees of the GUI. Examples: *Optimization->Constraints*, *or Environment->Celestial Bodies*, etc.
- *Shortcut Formatting* is used for keyboard shortcuts. Example: *Ctrl+G*, *Alt+F1*, etc.
- *Table Formatting* is used for row/column identifiers (headers) of a table in the GUI. Example: In the scenario modelling tree "Dynamics Configuration->Phases": *Use*, *Phase ID*, etc.
- *Tree Formatting* is used for main trees. Example: *Scenario*, *Optimization*, *Analyses*, *Results*, *Files*, etc.
- *Window Formatting* is used for all GUI windows. Examples: *Execution Log*, *Viewer*, *Add Analysis*, etc. windows.
  ```

2.9 Preferences

The *Preferences* window allows that global software and plotting settings are configured. It can be accessed by pressing the **Preferences** button in the lower right corner of the *Application menu* in the *ASTOS* GUI.

The left part of the window shows a navigation tree which is used to select a desired group of settings. On the right, a list of the settings belonging to the selected group is displayed. The settings are mainly self-explanatory and only a brief description is provided below.
Three buttons are present at the bottom right of the *Preferences* window to accept (**OK**), discard (**Cancel**) or simply apply (**Apply**) any modifications. Additionally, all settings can be reverted to the default status via the **Restore Defaults** button located at the lower left of window.
The setting groups in the navigation tree of the *Preferences* window are explained in more detail below.

General

The *General* setting are explained in the following.

- *Check automatically for software updates*: Activates a check whether the installed software version is the same as the latest available version on `https://www.astos.de`. In case a newer version is available, a pop-up message is displayed each time the software starts.
- *Create verbose output in Execution Log*: Causes that for each action (e.g. an optimization) performed by the user, the respective command is printed in the *Execution Log* as grey text. This is useful to find out the corresponding command to a specific action in order to be able to run this action on the command line.
- *Play sound after an optimization run is finished*: Optimization runs can be time-consuming. If this setting is active, after a task run is over a sound is played to inform the user about this.
- *Show introduction when opening a scenario*: If active, the software automatically opens a file called `scenario_intro.html` placed in the main scenario `gtp` folder. More details are provided in Section 2.1.1.2.
- *Let the operating system window handler decorate all windows and dialogs (requires restart)*: Paints the top bar of the windows using the operating system settings instead of the software color scheme.
- *Use internal browser to show html files (like introduction, summary, reports)*: If this setting is active, the built-in browser in the software is used to show `html` files instead of the system standard Internet browser (e.g. *Firefox*).
- *Keep NLP solver output files*: The native solver output files (e.g. `snopt.out`, `sos.out`, `worhp_restart.gz`) are usually deleted once the optimization run ends. This option prevents the deletion of these files if these are needed e.g. for debugging purpose.
- *Show again all 'Did you know?' info messages*: In several situations the *ASTOS* GUI shows pop-up messages containing useful information related to a certain task. Each of these pop-ups can be permanently discarded via a *Don't show it again* flag by the user. Clicking **Reset History** causes that all discarded pop-up messages are shown again.

Plot

Some of the *Plot* preferences are also available in the *Viewer* window menu (Section 2.2). The available settings in the *Preferences* windows are explained below.

- *Show the plot legend by default*: Adds a legend to all the plots in the tabs of the *Viewer* window. Legends are only present if they are necessary to distinguish different plotted lines (e.g. data from different sources).
- *Add header when printing plots*: Add the scenario name and time/date in the page header of a printed plot.
- *Show phase borders*: Highlight the transitions between scenario phases on the graphs in the *Viewer* window. If enabled, a square bracket is drawn on each data graph at the beginning (grey) and at the end (red) of each phase.
- *Display plots on dark background*: If active, a dark background is used for all plots. Otherwise a white background is used.
- *Use colored caption* in *2D Plot*: The different graphs in a 2D plot have different colours. If this flag is active, the graph colours are also used for the corresponding y-axis labels.
- *Use alternative line style scheme* in *Map Plot*: If this flag is inactive, the colour is used to identify the graphs in a single plot and the line stroke is used to identify the data sources in this plot. If the flag is active, these settings are inverted.
- *Default map background* in *Map Plot*: In this setting the user can choose the default background map in the *Map-Plot*, *3D Map-Plot* and *Satellite Plot* from a drop-down list.
- *Show complete history* in *Review Iteration Plot*: In the *Review Iterations* panel of the software (Section 6.12) the the user can review and visualize the evolution of the iteration values of quantities related to an optimization run. If this setting is disabled (default setting), only a reduced set of iterations is available (e.g. 10). If the setting is enabled, the complete iteration history is available for all quantities in the *Review Iterations* panel.

Report

These settings refer to the report generation located in the *Results* tree, *Reports* node (Section 8.4).

- *Add thousand separator to numbers*: If the setting is active, the number format used for output in *ASTOS* includes a thousand separator as defined by the *Used locale to format numbers* list.
- *Used locale to format numbers*: a drop-down list of locales defining the formatting scheme that applies to numbers, e.g. decimal separator.

Model Database

In the *Model Database* node settings the user can specify the location of the root folder of the model database. By default the model database is in the installation path of the software. The **...** button is used to browse to the desired location.

2.10 Data Input Fields and Tables

The *ASTOS* GUI provides a wide range of input options to set up user-defined data. The most common entities are described in this section together with examples to help the user defining the desired scenario.

After a description of the different input fields, several details are provided for data tables regarding general editing aspects, classification of tabular data and the possibility to introduce uncertainties or perturbations. Moreover, the following sections present a complete list of units implemented in the software and how variables can be used to reference multiple inputs to a central location. These variables can be used internally, for automatic processes (like batch modes) or to link *Excel* files and *SQL* databases.

2.10.1 Input Fields

Input fields are usually preceded by a descriptive *label*. They cover different types of user input:

- *Numeric values* are used to define scalar input data. Most input fields accept real values (e.g. mass of a component), some are restricted to integer values (e.g. the year for a calendar date).

> **Tip:** Default values of an input field (together with the associated unit) can be retrieved using the right-click menu (*Reset to Default Value*).

- *String* inputs are required to identify external files by providing their relative or absolute path (see e.g. Example 2 in Section 10.1.2.2), to specify descriptions (e.g. of scenarios, constraints or phases) and to specify identifiers (e.g. materials or vehicle elements).
- *Check markers* (box or circle shape) are commonly used to define the status of an entity, e.g. which actuator is used which assembly is jettisoned or which constraint is active in a specific phase. Some check marker groups allow multiple choices, others are restricted to a single choice (e.g. a constraint cannot be defined as initial and as final boundary in the same phase).

The *ASTOS* GUI automatically checks user inputs for consistency, i.e. an input field designed for the path of a required file accepts a valid path only; an input field designed for a floating number does not accept a generic string. Invalid inputs are usually indicated (highlighted by yellow background) already while editing an input field. More details about possible errors are provided in Chapter 12.

Not all input fields are immediately visible to the user. Various user options are initially set to *default* or are *disabled* and no user input is required. A left-click on these toggle buttons switches their state to *custom* or *enabled* respectively. In most cases, this reveals one or more additional input fields to be edited (e.g. for the *Spin state* in the configuration panel of *Celestial Bodies*).

Tip: Input fields can be linked to variables using the right-click menu. Refer to Section 2.10.4 for more details.

Apart from direct keyboard input to a selected (numerical value or string) input field, *ASTOS* also supports keyboard shortcuts for Copy and Paste operations in *Microsoft Windows* and *Linux* software. Possible key combinations are:

- *Ctrl+X* for cut
- *Ctrl+C* for copy
- *Ctrl+V* for paste

For *Mac OS* the Cmd key is used instead:

- *Cmd+X* for cut
- *Cmd+C* for copy
- *Cmd+V* for paste

Tip: Copy and paste functionality is also available from the right-click menu of (string and numerical value) input fields.

2.10.2 Profile Data Tables

Profile data tables are flexible data objects intended for modelling all data profiles derived from tabular data in the *Modelling* tree. The subsequent list gives an overview of the general properties of profile data tables. A complete description of all configuration options follows further below.

- Several interpolation and approximation methods are available for tabular data (e.g. linear or spline interpolation).
- One- or multi-dimensional tabular data can be handled. Two basic input modes are available from the *data source* drop-down menu:
  - *File*: import data from an external file (e.g. tabulator separated files)
  - *Local*: data profile is entered in the GUI manually by the user using e.g. keyboard
- Parameter uncertainties can be inserted using the *Perturbation (uncertainty)* panel.
- Profile data table objects can be referenced (i.e. linked to variables, see also Section 2.10.4) with a right-click on the icon  to the left of the panel label.

General items

The individual setup options of a profile data table are described in the following.

- *Angle of attack reverse lookup*
  This optional setting is only available for the coefficients of a tabular aerodynamics model (see section Tabular Aerodynamics in the Model Reference book). When *enabled* the setting *Degree of polynomial approximation* is available, specifying the degree of the polynomial created for the second independent variable (i.e. second column) of the profile data table which needs to be the *Angle of attack*. A polynomial will be evaluated for the angle of attack as function of the provided coefficient for reverse interpolation. This feature is required in particular for the attitude control law *Dynamic pressure* (please refer to section Dynamic Pressure in the Model Reference book).
- *Interpolation type*
  Defines how values between provided table data points are treated. Possible options are *Linear*, *Akima*, *Spline* or *General B Spline*. See Recommendations (Optimization Theory and Description of Methods) for more information about how to choose an interpolation type. Interpolated approximation - except for *General B Spline* - requires rectangular data.
- *Out of bounds action*
  Defines how function argument values which are outside the range covered by the tabular data are treated. Possible options are *Raise Exception*, *Nearest Value* (Default) or *Extrapolate*. The *Extrapolate* action follows the *Interpolation Type* set above (e.g. linear extrapolation). *Nearest Value* means that the function value is set to be equal to the function value for the closest function argument value in the table, i.e. either of the boundary values.
- *Ordinate data scale*
  Defines in which scaling the ordinate value is given and in which scaling it should be used for interpolation. The setting is only valid for 1-dimensional data. Possible options are *Linear to Linear* (default), *Linear to Log* or *Log to Log*. *Linear to Log* can be used if the data is given by pure function values, but the interpolation should be done more effectively with the logarithm of the values. *Linear To Log* is especially useful for atmospheric data, where density and pressure are exponential functions.
- *Scaling factor*
  Scales the values given in *Data* column. This factor can be used as a batch variable for Monte Carlo analysis (see Section 2.5 for more details) as well as a fast way to conduct a sensitivity analysis. The functionality is identical to the *Gain* entry of Section 2.10.2.3.
- *Data source*
  Defines the source of the tabular data. For a manual keyboard input of values, choose *Local*. *File* allows the import of data from a file.
- *Filename* (only available if *File* is chosen as *Data source*)
  Defines the location and name of the file used for the data import. Either the absolute or relative path w.r.t to the scenario directory (i.e.`example.gtp` folder) has to be provided. Alternatively the **...** button next to the *Filename* label can be used to browse the file location.
- *Data*
  The *Data* column is used to define the data to be interpolated. In case of an import from file, specify the column number that contains the respective data in the file.
- *Name*
  The *Name* row contains drop-down menus that can be open using a left-click. Use these menus to select the independent variable for the column (available options depend on the

type of the profile). The respective *Unit* cell below should be consistent with the selected *Name*.

- *Unit*
  The *Unit* row contains units strings. An entry is required for each independent variable and the interpolated data. Units must be given according to the quantities involved, a complete list can be found in Section 2.10.3.
- *Column*/*Identifier* (only available if *File* is chosen as *Data source*)
  The row containing *Column* or *Identifier* is only available in case *File* is selected as *Data source*. The two options can be switched by a left-click on the currently visible row label. They are used to map the data located in a file to columns in the profile table (for both data and the independent variable).

  - *Column* is used to determine the data location by integer numbers that correspond to the respective data column in the file (e.g. *2* maps the 2^{nd} column in the file to the selected table column).
  - *Identifier* is used to determine the data location by an identifier (string) present in the import file that correspond to the respective data column in the file.

Note: Using *Identifier* is mandatory in case the import file is in matrix format (see Section 10.1.2.2 and following sections).

Tip: *ASTOS* automatically removes headers (column names) during the file import.

A dedicated toolbar is available above the table to preview the data as well as to add, delete and modify columns; all individual icons are described in Section 2.10.2.1.

Proper definition of independent variables

Each independent variable has to be either monotonically increasing or decreasing. When using a file created by *ASTOS* subsequent rows with identical independent variable values can be present. In cases the ordinate values are identical, the duplicated entry is ignored. However, if the data values are not identical and linear interpolation is used, the second abscissa value is increased by 10^{-8}. This allows to read in e.g. jumps in a control between two phases. Normally this jump occurs at one time point.

2.10.2.1 General Table Editing

This section provides information on the general aspects when editing tabular data in the GUI, i.e.

- Adding data to Table
- Modifying columns
- Selecting the physical dimension

- Defining input units
- Preview data
- Tracking table modifications

Adding data to a table

How data is mapped to the cells of a table is depending on the selected *Data source*. Two options exist:

- The *Local* setting enables the user to enter data values manually into the table cells (below the *Unit* row). Entering the values can be realized either by direct keyboard input to a selected cell or by using the copy and paste functionality provided by *Microsoft Windows* and *Linux*. The following shortcuts for the cut, copy and paste are supported:

    - *Ctrl+X* for cut
    - *Ctrl+C* for copy
    - *Ctrl+V* for paste

    On *Mac OS* the Cmd key instead of the Ctrl key shall be used:

    - *Cmd+X* for cut
    - *Cmd+C* for copy
    - *Cmd+V* for paste

    Selection of multiple cells in the table can be realized by dragging the mouse over several table cells or by pressing the Shift key together with the cursor keys.
    It is possible to paste only a single value (and no multiple cells) which has been copied e.g. from a *Microsoft Excel* spreadsheet into an *ASTOS* table. In case the user desires to use multiple cells from a *Microsoft Excel* spreadsheet, this can be done by creating a database variable linked to a spreadsheet and then indicating from which area inside the spreadsheet the data shall be adopted. The procedure how to do this is explained in Section 2.10.4.2.
- If *File* is selected as *Data source*, a *Filename* of an import file including its path can be specified (directly as a string or using the **...** button to browse the file system). In this case, the data properties, i.e. physical dimensions, units and identifiers for the data location in the file have to be provided by the user. A detailed description of importing data from a file can be found in Section 10.1.2.

Modifying columns

Columns can be modified using the toolbar above a table. The meaning of the icons from left to right is (in case *File* is selected as *Data source*, only icons marked with an * are available):

- Move column to left
- Move column to right
- Insert column at left from selection*
- Insert column at right from selection*

- Remove selected column*
- Insert row above selection
- Insert row below selection
- Remove selected row

Selecting the physical dimension

The most important setting is to define the dimension of the independent variables. A left-click on a cell in the *Name* row opens a drop-down list with all possible dimension types a the specific model.

Defining input units

The *Unit* row allows to define any kind of *ASTOS* unit and assign it to a column. If the selected unit is not compatible with the physical dimension selected in *Name* row above, an error message is displayed. A complete list of units supported by *ASTOS* is provided in Section 2.10.3.

Preview data

The **Preview** button (on the right of the table toolbar) opens a separate window with the plot of the data inserted by the user. The main purpose is to check the correctness of the inserted data before running a simulation. The interaction with the data preview is performed via mouse actions while the plot behavior is identical to the one in the *Viewer*.
In case of a 2-D plot:

- the right mouse button can be used to draw a box to zoom into it,
- the left mouse button can be used to pan the view.

In case of a 3-D plot:

- the right mouse button can be used to pan the view,
- the left mouse button can be used to rotate the view,
- the mouse wheel can be used to zoom-in/out.

Tracking table modifications

In general, a focus listener detects if the table loses focus while a cell is edited by the user. However, in some cases this is not possible, e.g. when selecting another element in the *Modelling* tree while a table cell is edited. In this case, the currently edited cell value is not applied.
Hence take care of the following advices:

> **Tip:** Finalize cell editing by selecting a different element (not background) in the same configuration panel (not by selecting a tree, toolbar or menu item).

> **Tip:** Finalize cell editing with the Return/ENTER key.

2.10.2.2 Classification of Data and Approximation Methods

Table 2.1 shows the available approximation and interpolation methods in *ASTOS* depending on the number of independent variables and the required data format.

"Rectangular data" means that the array of an independent variable has always the same length as the provided (multi-dimensional) data values, i.e. a band structure is usually not supported. These arrays are neither fixed in length nor in the values in case of non-rectangular data. See subchapters of Section 10.1.2 for more information and examples on input formats of tabular data.

Table 2.1: Interpolation type according to the number of independent variables

Interpolation Type	**Rectangular Data**	**No. Indep. Variables**					
		1	**2**	**3**	**4**	**5**	**n**
Linear	Rectangular	X	X	X	X	X	
Akima	Unessential	X					
Spline	Rectangular	X	X				
General B-Spline	Also non-rectangular	X	X	X	X	X	X

Table 2.2 shows the required data structure depending on the number of independent variables. The GUI table generally has the same structure as an external file with columns. n-dimensional data must be provided as B-Spline coefficients.

Table 2.2: Data definition according to the number of independent variables

No. Indep. Vars	**Columns (Local or from External File)**	**Matrices (External File**
1	2 columns (1 variable, 1 data)	
2	3 columns (2 variables, 1 data)	2D-matrix
3	4 columns (3 variables, 1 data)	3D-matrix
4	5 columns (4 variables, 1 data)	
n	1 column	

Note: The *General-B-Spline* requires *File* as *Data source*.

2.10.2.3 Perturbation (uncertainty)

Depending on the analysis, it is quite common that the data available is subject to a certain level of uncertainty, i.e. the known values are not representative for all real world situations. For example, the atmospheric characteristics (density, pressure and speed of sound) are known for a certain position at a certain time, but there is always a variation between the model values and the real ones.

Uncertainty is theoretically present in all the data that can be used in the GUI: values and profiles. In case of single values, it is easy to link them to batch variables and use the *Batch-Mode Inspector* to access and modify them. Please refer to Section 2.5 for more details. In case of profiles, a common scaling factor can be treated as a single value and modified via the *Batch-Mode Inspector*. In case the uncertainty is not constant throughout the profile, several uncertainty profiles can be defined.

The *Perturbation (uncertainty)* settings (*disabled* by default) are available for all profile entities in the configuration panel. If *enabled*, two additional settings can be specified in order to define a complex perturbed profile: *Bias* and *Gain*. They can be *Defined by* a single *Value* only or by a *Profile*. In the latter case, the *Perturbation (uncertainty)* section is extended by one interpolation table (for *Bias* and *Gain* respectively) that allows the definition of the uncertainty profile for the respective entity.

Effect on nominal values

The perturbed value/profile is computed according to the following expression:

$$\mathrm{PerturbedValue} = Bias + \mathrm{NominalValue} \cdot Gain \tag{2.1}$$

where the nominal values are taken from the default interpolation profile. A *Bias* of 0 and a *Gain* of 1 would thus represent the nominal profile again.

Tip: In case a Monte Carlo simulation with known 1-sigma deviations for the nominal values shall be run, it is suggested to set the *Bias* as *Profile*, insert the 1-sigma values in the *Profile* and use its *Scaling factor* as batch mode variable (see Section 2.5 for more details on batch mode variables).

2.10.3 Units

Internally, all computations are carried out in in SI units. However, user inputs can be specified in any unit which is supported by the model library. More than 100 unit definitions are available for that purpose (see Section 2.10.3.2). Also composite units are possible.

Most input fields are followed by a unit selection drop-down list which contains at least one predefined unit for the related quantity. It is also possible to create a custom units (see Section 2.10.3.1).

ASTOS unit strings use a style that avoids potential confusion. Examples of valid units are "Kilo-Joule", "Meter*Kelvin/Second", "Nano-Meter/Second**2" or "Newton/(Kilo-Meter*Kelvin)". Abbreviated unit strings are displayed in case of plot labels and report values. All input fields use the long style shown above.

2.10.3.1 Create Custom Units

If a required unit is not available in a unit selection drop-down list of an input field, it can be defined or composed manually by choosing the last entry *Custom...*.
This opens the *Change Unit* dialog window. For tables that contain a *Units* row, the same dialog can be accessed by a left-click on the unit icon ( ) in a related table cell. Two general options are available to change a unit.

Select a unit from the proposed list

The *Proposed list* directly provides common alternative units of the same category. If available, the desired unit can be selected from the list and the dialog can be closed using the **Ok** button. The new unit is now present in the drop-down list in case of an input field or applied to the *Units* cell in case of a table.

Create a user-defined unit

It is also possible to build up a *User-defined unit* from scratch. The following setup options are available:

- *Unit group* specifies the category of a unit element, e.g. *Length*.
- *Prefix & unit* are drop-down lists to specify an available unit of that *Unit group* (e.g. *Meter*) which can optionally be preceded by a metric system prefix (e.g. *Centi-*).
- A panel with buttons to add or remove elements of a composed unit. The functionality is described below.

To **add an element** to a composed unit at the desired location (numerator or denominator), use the respective **+** button above or below the fraction bar. Added elements then appear in the *Short unit string preview* panel.
To **remove an element** select it with a left-click in the *Short unit string preview* panel and use the respective **-** button above or below the fraction bar.
To **increase/decrease an exponent of an existing element**, select it with a left-click and use the respective **+/-** button again.

Tip: Selected elements of a composed unit are indicated with a red box in the *Short unit string preview* panel.

Note: A user-defined unit can be composed of elements from different unit categories (i.e. length, mass, etc.), but the result has to be compliant with the quantity to be described.

The *SI view* panel displays the conversion factor of the user-defined unit with respect to the related SI unit of the quantity. In case the resulting unit does not match, "[incompatible unit]" is displayed.

Example: Recreating [Newton] from fundamental SI units

Although *Newton* is already defined in the *Unit group Force*, it could also be "rebuilt" from its SI components (kgm/s^2) as follows:

1. Select *Create a user-defined unit*.
2. Choose *Mass* from the *Unit group* list, then *Kilogram* from the second *Prefix & unit* list. No prefix is required from the first list.
3. Click the **+** button above the fraction bar to add the element to the numerator.
4. Choose *Length* from the *Unit group* list, then *Meter* from the second *Prefix & unit* list. No prefix is required.
5. Click the **+** button above the fraction bar to add the element as factor to the numerator.
6. Choose *Time and Frequency* from the *Unit group* list, then *Second* from the second *Prefix & unit* list. Again, no prefix is required.
7. Click the **+** button below the fraction bar **twice** to add the squared element to the denominator.

2.10.3.2 Available Units

Standard SI units are available in the form of predefined unit constants listed in Table 2.3 .

Table 2.3: List of base SI units

SI Unit Constants
Meter
Kilogram
Second
Radian
Steradian
Ampere
Kelvin
Mole
Candela

All following units - sorted by category - are available in the software. Supported metric system prefixes are summarized in Table 2.26 .

Time Unit Keywords (SI-Unit: Second):

Table 2.4: List of available units for time

Keyword	Short	Conversion
Century	—	100.0 * Year
Day	d	24.0 * Hour
Decade	—	10.0 * Year
Fortnight	—	2.0 * Week
Hour	h	60.0 * Minute
Juliancentury	—	100.0 * Julianyear
Julianyear	—	365.25 * Day
Millenium	—	1000.0 * Year
Minute	min	60.0 * Second
Month	—	Year / 12.0
Second	s	—
Siderealyear	—	365.25636 * Day
Tropicalyear	—	365.24219 * Day
Week	—	7.0 * Day
Year	—	365.0 * Day

Frequency Unit Keywords:

Table 2.5: List of available units for frequency

Keyword	Short	Conversion
Hertz	Hz	1.0 / Second
Bequerel	Bq	1.0 / Second

Length Unit Keywords (SI-Unit: Meter):

Table 2.6: List of available units for length

Keyword	Short	Conversion
Angstrom	A	1.0e-10 Meter
AstronomicalUnit	AU	1.49597870691e+11 Meter
Bolt	—	40.0 Yard
Cable	—	720.0 Foot
Caliber	—	1.0e-2 Inch
Chain	—	66.0 Foot
Cubit	—	18.0 Inch
Ell	—	45.0 Inch
Fathom	—	6.0 Foot
Feet	ft	Foot
Foot	ft	0.3048 Meter
Furlong	—	220.0 Yard

Keyword	Short	Conversion
Hand	—	4.0 Inch
Inch	in	2.54e-2 Meter
League	—	3.0 Mile
Lightyear	ly	9.460528e+15 Meter
Link	—	7.92 Inch
Meter	m	—
Micron	—	1.0e-6 Meter
Mil	—	1.0e-3 Inch
Mile	mi	1.609344e+3 Meter
Nauticalmile	nmi	1.852e+3 Meter
Parsec	pc	30.856776e+15 Meter
Rod	—	5.5 Yard
Rope	—	20.0 Foot
Skein	—	360.0 Foot
Span	—	9.0 Inch
Stadion	—	622.0 Foot
Stadium	—	202.0 Yard
Surveymile	—	320.0 Rod
Yard	yd	0.9144 Meter

Area Unit Keywords:

Table 2.7: List of available units for area

Keyword	Short	Conversion
Acre	—	0.404686 * Hectare
Are	—	Acre / 40.0
Barn	—	1.0e-28 * Meter**2
Hectare	ha	1.0e+4 * Meter**2
Rood	—	Acre / 4.0
Section	—	Mile**2
Township	—	36.0 * Section

Volume Unit Keywords:

Table 2.8: List of available units for volume

Keyword	Short	Conversion
Bag	—	3.0 * Bushel
Barrel	—	0.1590 * Stere
Boardfoot	—	144.0 * Inch**3
Bucket	—	4.0 * Gallon

Keyword	Short	Conversion
Bushel	bu	4.0 * Peck
Butt	—	126.0 * Gallon
Cord	—	128.0 * Foot**3
Cup	—	0.5 * Pint
Drop	—	0.3e-7 * Stere
Ephah	—	10.0 * Omer
Fifth	—	0.8 * Quart
Firkin	—	9.0 * UKGallon
Fluiddram	—	0.125 * Fluidounce
Fluidounce	fl oz	Pint / 16.0
Gallon	US gal	3.78541 * Liter
Gill	—	0.25 * Pint
Hogshead	—	0.5 * Butt
Jigger	—	1.5 * Shot
Last	—	2909.414 * Liter
Liter	l	1.0e-3 * Meter**3
Magnum	—	2.0 * Quart
Minim	—	Fluidounce / 480.0
Omer	—	0.45 * Peck
Peck	—	8.810 * Liter
Pint	pt	Gallon / 8.0
Pony	—	0.5 * Jigger
Puncheon	—	84.0 * Gallon
Quart	qt	2.0 * Pint
Registerton	reg ton	100.0 * Foot**3
Seam	—	8.0 * Bushel
Shot	—	Fluidounce
Stere	—	Meter**3
Tablespoon	—	4.0 * Fluiddram
Teaspoon	—	Tablespoon / 3.0
Tun	—	4.0 * Hogshead
UKGallon	UK gal	4.54609 * Liter
UKPint	pt	UKGallon / 8.0
Winebottle	—	0.7576778 * Liter

Mass Unit Keywords (SI-Unit: Kilogram):

Table 2.9: List of available units for mass

Keyword	Short	Conversion
Assayton	—	29.167 * Gram
Bale	—	500.0 * Pound
Carat	—	0.2 * Gram
Cental	—	100.0 * Pound
Drachma	—	4.2923 * Gram
Grain	—	64.799e-3 * Gram
Gram	g	1.0e-3 * Kilogram
Kilogram	Kg	—
Libra	—	325.971 * Gram
Metricton	—	Tonne
Mina	—	0.9463 * Pound
Obolos	—	715.38e-3 * Gram
Ounce	oz	28.34952 * Gram
Pennyweight	—	1.555 * Gram
Pondus	fl oz	0.71864 * Pound
Pound	lb	0.45359237 * Kilogram
Shekel	—	14.1 * Gram
Shortton	sh tn	2000.0 * Pound
Slug	—	14.5939029372 * Kilogram
Stone	—	14.0 * Pound
Talent	l	60.0 * Mina
Ton	US tn	907.1847 * Kilogram
Tonne	t	1.0e+6 * Gram
Troyounce	—	31.103 * Gram
UKTon	UK tn	2240.0 * Pound
Wey	—	252.0 * Pound

Speed Unit Keywords:

Table 2.10: List of available units for speed

Keyword	Short	Conversion
Knot	kt	Nauticalmile / Hour

Gravity Unit Keywords:

Table 2.11: List of available units for gravity acceleration

Keyword	Short	Conversion
Gal	—	1.0e-2 * Meter/Second**2

Keyword	Short	Conversion
Gravity	g	9.80665 * Meter/ Second**2

Force Unit Keywords:

Table 2.12: List of available units for force

Keyword	Short	Conversion
Dyne	—	1.0e-5 * Newton
Kilogramforce	—	9.80665 * Newton
Newton	N	Kilogram * Meter / Second**2
Poundal	pdl	0.138255 * Newton
Poundforce	lbf	4.448222 * Newton
Tonforce	—	9964.02 * Newton

Pressure Unit Keywords:

Table 2.13: List of available units for pressure

Keyword	Short	Conversion
Atmosphere	atm	0.101325e+6 * Pascal
Bar	bar	1.0e+5 * Pascal
Barye	—	0.1 * Pascal
Pascal	Pa	Newton / Meter**2
Poundsquarefoot	psf	47.8802589 * Pascal
Poundsquareinch	psi	6.894757e+3 * Pascal
Torr	—	0.13322e+3 * Pascal

Energy Unit Keywords (SI-Unit: Joule):

Table 2.14: List of available units for energy

Keyword	Short	Conversion
BTU	BTU	1055.06 * Joule
Calorie	cal	4.1868 * Joule
Erg	—	1.0e-7 * Joule
Joule	j	Newton * Second
Rydberg	—	2.1799e-11 * Erg
Therm	—	1.0e+5 * BTU

Specific Energy Unit Keywords (SI-Unit: Grey):

Table 2.15: List of available units for specific energy

Keyword	Short	Conversion
Grey	—	Joule / Kilogram

Power Unit Keywords (SI-Unit: Watt):

Table 2.16: List of available units for power

Keyword	Short	Conversion
Chevalvapeur	—	735.499 * Watt
Horsepower	hp	745.700 * Watt
Watt	W	Joule / Second

Temperature Unit Keywords (SI-Unit: Kelvin):

Table 2.17: List of available units for temperature

Keyword	Short	Conversion
Kelvin	K	—
Rankine	°R	0.5556 Kelvin

Viscosity Unit Keywords:

Table 2.18: List of available units for viscosity

Keyword	Short	Conversion
Poise	—	0.1 * Pascal * Second
Reyn	—	6.89476e+4 * Poise
Stokes	—	1.0e-4 * (Meter**2)/ Second
Rhes	—	1.0 / Poise

Electrical and Magnetical Unit Keywords:

Table 2.19: List of available units for Electrical and Magnetic entities

Keyword	Short	Conversion
Ampere	A	—
Coulomb	C	Ampere * Second
Farad	F	Ampere * Second / Volt
Henry	H	Volt * Second / Ampere
Ohm	Ω	Volt / Ampere
Siemens	S	1.0 / Ohm
Tesla	T	Volt * Second / Meter^2
Volt	V	Watt / Ampere
Watt	W	Joule / Second
Weber	Wb	Volt * Second

Light related Unit Keywords (SI-Unit: Candela):

Table 2.20: List of available units for Light

Keyword	Short	Conversion
Candela	cd	—
Lumen	lm	Candela * Steradian
Lux	lx	Lumen / Meter^2

Radioactivity Unit Keywords:

Table 2.21: List of available units for Radioactivity

Keyword	Short	Conversion
Bequerel	Bq	1.0 / Second
Sievert	Sv	Joule / Kilogram

Biochemical Unit Keywords:

Table 2.22: List of available units for Biochemical

Keyword	Short	Conversion
Katal	kat	Mole / Second

Molecular Weight Unit Keywords (SI-Unit: Mole):

Table 2.23: List of available units for Molecular Weight

Keyword	Short	Conversion
Mole	—	—

Angular Units Keywords (SI-Unit: Radian):

Table 2.24: List of available units for angular units

Keyword	Short	Conversion
Degree	deg	Pi / 180.0 * Radian
Radian	rad	—

Solid angle Units Keywords (SI-Unit: Steradian):

Table 2.25: List of available units for solid angle units

Keyword	Short	Conversion
Steradian	sr	—

Unit Prefixes:

Table 2.26: List of available unit prefixes

Yocto	Zepto	Atto	Femto
Pico	Nano	Micro	Milli
Centi	Deci	Deca	Hecto
Kilo	Mega	Giga	Tera
Peta	Exa	Zetta	Yotta

2.10.4 Variables

A variable represents a number or a table of numbers that can be linked to a numeric input field or table. *ASTOS* distinguishes between three kinds of variables:

- Static Variables
- Database Variables
- Batch Variables

Links to a variable help to reduce redundant inputs (static variables) or can be used to access data from external sources (database variables). Multiple links can be established to a single variables, but a variable can only exist once as is defined by a unique identifier. The *Batch-Mode Inspector* (see Section 2.5) only has access to batch variables.

All defined variables are shown in the *Variables* tree. *Static Variables* and *Database Variables* are saved by default in `model/variables.xml`, *Batch Variables* are saved in `batch/homotopy.xml`. The right-click of an existing variable in the tree provides the following options:

- *Delete* removes the selected variables and related links and opens the *Delete Variable* dialog. **Keep** assigns the current variable value to all linked inputs fields before deleting the link. **Restore** restores the original input field value(s) which have been present before the link to the variable was created.

  Note: Restore is only possible within a session, i.e. not after closing *ASTOS* or loading another scenario.

- *Rename* renames the selected variable.
- *Linked by* (not available for *Batch Variables*) shows all objects linked to the variable and allows a quick goto for all objects.

Tip: The name of a variable can be copied in memory with *Ctrl+ C*; this can be useful when the variable name has to be inserted in the *Batch-Mode Inspector*.

2.10.4.1 Static Variables

A static variable can be used to reduce redundant inputs (i.e. in case a certain value is required in many different input fields) or as a collection of important numbers.

Create a static variable

To create a static variable, open the right-click menu of input field or table (profile data table, see section Section 2.10.2). The menu options depend on the type of the input field:

- *Create 'Floating Point Value' Variable and Link* is available for all numerical input fields that are linked to a unit. This option links only the numerical value to the static variable without the unit.
- *Create 'Unit Value' Variable and Link* is only available for numerical input fields that are linked to unit, but in this case both numerical value and unit are linked to the static variable.
- *Create Variable and Link* is available for dimensionless numerical input fields and profile data tables. In the latter case, the complete table will be linked to the variable, i.e. it is not possible to link an individual cell only.

Independent of the selection, the *Create Variable and Link* dialog opens. Here, a unique *Variable name* has to be specified and the type of the variable needs to be selected (*Static Variable* in this case). Furthermore, *Target file* (located in the scenario `.gtp` folder) and variable *Type* are displayed for information. **Create** confirms the setup and the variable is now present in the *Static Variables* branch of the *Variables* tree.

Link to an existing static variable

To create a link to an existing static variable, open the right-click menu of a numeric input field or table (profile data tables, see section Section 2.10.2). The menu options depend on the type of the input field:

- *Link to 'Floating Point Value' Variable* is available for numerical input fields which are followed a unit list. This option links only the numerical value to the static variable without the unit information.
- *Link to 'Unit Value' Variable* is also available for numerical input fields that are linked to unit, but in this case both numerical value and unit are linked to the static variable.
- *Link to Variable* is available for dimensionless numerical input fields and profile data table. In the latter case, the complete table will be linked to the variable, i.e. it is not possible to link an individual cell only.

Independent of the selection, the *Link To Variable* dialog opens. Here, first a *Variable type* (*Static Variable* in this case) has to be selected. The list below then shows all compatible existing variables. The selection has to be confirmed with the *Link* button.

Note: Only compatible variables i.e. of variables of the same type and format are shown in *Link To Variable* (e.g. static variables with matching units).

Modifying variable values and links

As soon as an input field or table is linked to a variable, the respective field or table is greyed out and locked for modifications, i.e. the respective value(s) can only be edited from the *Variables* tree. A quick goto link to the referenced variable is available from the right-click menu (*Goto Link '[Variable Name]'*).
The right-click menu option *Remove Link* can be used to break the link to a variable without affecting the variable itself. In this case, the original value of the object is restored.

2.10.4.2 Database Variables

A database variable can be used to read numeric data from an external source (i.e. *MySQL* database or *Microsoft Excel* file). To set up database variables, one or more connections to a database need to be present in the scenario (see Section 10.3 on how to connect databases). The data are stored in the *Variables.xml* file, so the origin of the data is not required during the normal software operation.

Create a database variable

To create a database variable, open the right-click menu of input field or table (profile data tables, see section Section 2.10.2). The menu options depend on the type of the input field:

- *Create 'Floating Point Value' Variable and Link* is available for numerical input fields which are linked to a unit. This option links only the numerical value to the static variable without the unit.
- *Create Variable and Link* is available for dimensionless numerical input fields and profile data tables. In the latter case, the complete table will be linked to the variable, i.e. it is not possible to link an individual cell only.

Note: It is not possible to link units to database variables.

Independent of the selection, the *Create Variable and Link* dialog opens. Here, a unique *Variable name* has to be specified and the type of the variable needs to be selected (*Database Variable* in this case). Some additional configuration settings are required:

- *Type* defines the database connection, i.e. *Excel File* or *MySQL Database*.
- *DB Connection ID* contains a list of identifiers for database connection defined in the scenario (see Section 10.3).
- *Target file* provides the file name where the variable data is saved (in the scenario `.gtp` folder).

Note: At least one database connection needs to be defined (see Section 10.3) before database variables can be created.

Create confirms the setup and the variable is now present in the *Database Variables* branch of the *Variables* tree.

Link to an existing database variable

To create a link to an existing database variable, open the right-click menu of a numeric input field or table (profile data table, see section Section 2.10.2). The menu options depend on the type of the input field:

- *Link to 'Floating Point Value' Variable* is available for numerical input fields which are followed a unit list. This option links only the numerical value to the static variable without the unit information.
- *Link to Variable* is available for dimensionless numerical input fields and profile data tables. In the latter case, the complete table will be linked to the variable, i.e. it is not possible to link an individual cell only.

Independent of the selection, the *Link To Variable* dialog opens. Here, first a *Variable type* (*Database Variable* in this case) has to be selected. The list below then shows all compatible existing variables. The selection has to be confirmed with the *Link* button.

Note: Only compatible variables i.e. of variables of the same type and format are shown in *Link To Variable*.

Modifying links

As soon as an input field or table is linked to a variable, the respective field or table is greyed out and locked for modifications, i.e. the respective value(s) can only be edited from the *Variables* tree. A quick goto link to the referenced variable is available from the right-click menu (*Goto Link '[Variable Name]'*.
The right-click menu option *Remove Link* can be used to break the link to a variable without affecting the variable itself. In this case, the original value of the object is restored.

Configuration of database variable

To configure a database variable, select the respective entry in the *Database Variables* branch of the *Variables* tree. The configuration panel provides the following information and settings:

- *Identifier* represents the name/ID of the variable.
- *Type* shows variable type.
- *File* shows file target where the variable is saved.
- *DB connection ID* allows a quick switch to any other database connection of the same type.
- *SQL query* or *Excel query* is the query to the database or *Excel* file to fetch a new value, see Section 2.10.4.2.1 and Section 2.10.4.2.2 for more details.
- **Update Value(s)** triggers a manual update of the linked value(s).
- *Automatic update* triggers an automatic update of the value before it is accessed by *ASTOS*, e.g. before a simulation or optimization process is started (*disabled* by default).
- *Last update* shows the date and time of the last update process of the value(s).

Below the user has the possibility to change the value(s) also manually (this has no effect, when *Automatic update* is *enabled*!). This change will affect only the numeric data stored in the *Variables.xml* file; the origin of the data is not changed.

2.10.4.2.1 SQL Query

An *SQL* `SELECT` statement is used to fetch the data from a database table which returns data in the form of a result table. These result tables are called result-sets. The result-set should have a single value if the variable has the type *Floating Point Value* otherwise it uses only the first value of the set. In case of profile data tables, the result-set requires multiple values which have to be in line with the number of columns and rows of the linked table.

Note: The *SQL* statement has to start with *SELECT* and the result-set has to return a number or set of numbers.

Examples

Floating point value variable:

- `SELECT value1 FROM table_name`

Profile data table variables:

- `SELECT * FROM table_name`
- `SELECT column1, column2, columnN FROM table_name`

2.10.4.2.2 Excel Query

A *Microsoft Excel* file contains worksheets with a collection of cells that contain the data. To fetch values from cells of a worksheet, the name of the worksheet and the cell references have to be provided. The syntax is similar to an absolute cell reference in *Microsoft Excel*, i.e. it consists of:

1. Name of the worksheet
2. Delimiter "!"
3. Cell identifiers

The syntax is: `[worksheet]![cell]` or `[worksheet]![cell]:[cell]`

Note: The query does not have to start with = as usually in *Microsoft Excel*.

Examples

- `my_worksheet!A1` or `my_worksheet!c1r1` for a scalar floating point value variable from the first cell
- `my_worksheet!A1:C10` or `my_worksheet!c1r1:c3r10` for a profile data table variable with values from the first 3 columns and first 10 rows
- `my_worksheet!A1:A10;my_worksheet!C1:C10` or `my_worksheet!c1r1:c1r10;my_worksheet!c3r1:c3r10` for a profile data table variable with values from the first column and from the third one (non consecutive range) and first 10 rows.

Note: In case of an `A1` syntax the column identifier must be given in capital letters; in case of `c1r1` syntax `c` and `r` are in lower case letters.

In case the *Microsoft Excel* cells contain formulas, the number of digits read could be different from the number of digits shown in *Microsoft Excel*.

2.10.4.3 Batch Variables

A batch variable is needed to create a reference for the *Batch-Mode Inspector*. Refer to Section 9.1 for more details about batch operations.

Create a batch variable

To create a batch variable, open the right-click menu of a numerical input field and select *Create 'Floating Point Value' Variable and Link* or *Create Variable and Link*.

Note: It is not possible to link units or tables to batch variables.

This opens the *Create Variable and Link* dialog where first a unique *Variable name* has to be specified and the type of the variable needs to be selected (*Batch Variable* in this case). **Create** confirms the setup and the variable is now present in the *Batch Variables* branch of the *Variables* tree.

Link to an existing batch variable

To create a link to an existing batch variable, open the right-click menu of a numeric input field and choose *Link to 'Floating Point Value' Variable* or *Link to Variable*. The *Link To Variable* dialog opens where first a *Variable type* (*Batch Variable* in this case) has to be selected. The list below then shows all compatible existing batch variables. The selection has to be confirmed with the *Link* button.

Modifying variable values and links

As soon as an input field is linked to a variable, the respective field is greyed out and locked for modifications, i.e. the respective value(s) can only be edited from the *Variables* tree. A quick goto link to the referenced variable is available from the right-click menu (*Goto Link '[Variable Name]'*.

The right-click menu option *Remove Link* can be used to break the link to a variable without affecting the variable itself. In this case, the original value of the input field is restored.

Tip: The name of a variable can be copied in memory with *Ctrl*+ *C*; this can be useful when the variable name has to be inserted in the *Batch-Mode Inspector*.

3 ASTOS Scenario Overview

This chapter shall give an overview on the components of an *ASTOS* scenario and shall give an insight into the workflow. The settings and definitions that are part of a scenario are grouped into the following categories:

- Modelling
- Analyses
- Variables
- Optimization
- Results
- Files

These categories can be found as tree header on the left of the main *ASTOS* window. Each category is explained in the following and detailed in dedicated chapters.

The *Modelling* tree is used to define constellations, vehicles, the vehicle dynamics, ground stations, points and areas of interest and also the environment. A vehicle (as well as a ground station) may consist of several elements like components (structural elements), actuators, sensors and also characteristics (e.g. aerodynamics).

Apart from the normal simulation of the scenario *ASTOS* provides a set of post-simulation analyses. Once added to the scenario those analyses can be found in the *Analyses* tree. Typically those analyses are separated from a standard simulation as they are time-consuming and not necessarily required after every intermediate simulation.

It is possible to link most of the input parameters of models to variables. Those variables might refer to static values and shall simplify the input of frequently used values. Other types of variables can link *ASTOS* to databases and *Excel* sheets or they serve as interface to the built-in batch-processing tool of *ASTOS*. Defined variables are listed in the *Variables* tree.

The *Optimization* tree summarizes all settings and tools that are related to trajectory or design optimisation. Constraints and cost functions are defined here. Also grid settings and solver parameters can be specified in this category.

The results of simulations, analyses and optimizations can be displayed in terms of plots, animations and reports. Alternatively, it is possible to export results into third-party formats. Settings and tools related to plotting, reports or exports can be found in the *Results* tree.

The *Files* tree lists all configuration files linked to the scenario. This is especially useful in case models are defined outside the scenario folder, which is typically the case if a central database was established that contains predefined element models used by several scenarios. Via the *Files* tree it is also possible to mark a file write-protected to avoid accidental changes.

The ASTOS scenario on file system level

In the following it is explained how *ASTOS* stores scenario configurations and how it saves results, which is required to be understood in order to successfully work with *ASTOS*.

On file system level, an *ASTOS* scenario consists of a folder whose name ends with `.gtp`. This folder contains several files and subfolders. The configuration defined in the *Modelling* tree is stored in several `XML` files located in the `model/astos` subfolder. Those files are listed in the *Files* tree (and maybe further files stored otuside the `gtp` folder). Simulation settings as well as grids and (optimal) optimization variable values and bounds are stored in the `input.tops` files located in the root `.gtp` folder. The simulation and analysis results are stored in `STRUCT` files (the default is `simulation.struct`), which are by default located in the `integ` subfolder. The `gismo` subfolder contains files that are used by the optimization iteration history graphs. These files contain key parameters of the optimization scenario for stored iterations at a user-defined frequency. The `plot` folder contains configuration files for the diagrams and plots shown in the *Viewer* window and the settings made in *Astroview*. Automatically created reports are stored in a subfolder of the `reports` folder and named according to the report. In the `geometry` folder all 3D models of spacecrafts and their components shall be stored. The geometry folder might contain also a subfolder `automatic`. This folder is dedicated to geometry files that are automatically created by *ASTOS*. The batch folder contains the configuration files for the *Batch-Mode Inspector* as well as the `homotopy.xml` file. This file contains all defined batch variables and their values. The `exports` folder is the default target for exported simulation and analysis data.

Workflow

In *ASTOS*, settings made in the GUI typically have to be saved before they become effective. Furthermore, it is required to run a simulation, analysis, initialisation or optimization action to obtain results. The workflow of *ASTOS* depends on the application. If the scenario shall be optimized (the **Optimization** feature must be available, i.e. present in the configuration panel of the root node of the *Modelling* tree and *enabled*), then a (minimalistic) workflow consists of the following steps:

1. Configure the scenario, grids and optimization settings
2. Initialize the scenario (this creates the initial guess and creates the required grid data, both stored in the `input.tops` file)
3. Optimize the scenario (this step does not necessarily create a simulation output file)
4. Simulate the scenario (this step creates the simulation output using controls and optimizable parameters from the previous optimization)
5. Run further post-simulation analyses (optional)
6. Inspect the results using *Viewer*, *Astroview* and reports

It has to be highlighted that above workflow just shows a very basic procedure. Detailed information on how to use *ASTOS* as an optimization tool can be found in Chapter 6. In case optimization techniques shall not be applied to the scenario (*Optimization* should be *disabled* in the *Modelling* tree root node if available), the workflow is simpler than with optimization involved:

1. Configure the scenario
2. Simulate the scenario
3. Run further post-simulation analyses (optional)
4. Inspect the results using *Viewer*, *Astroview* and reports

In both cases a more complex workflow might be applicable, e.g when using the batch processing capabilities of *ASTOS* which are described in Chapter 9.

4 Modelling of Scenarios

The *Modelling* tree contains a list of all model components which describe vehicles and their environment. This list includes e.g. sensors, actuators, structural and aerodynamics model components of a vehicle as well as the default environment: celestial body and atmospheric properties.

Model components defined in *Vehicle Parts & Properties* are the building blocks from which a vehicle model is made of. In *Vehicles & POIs Definition*, the vehicle is assembled from these templates and used according to the settings of *Dynamics Configuration* (see Section 4.5) in a simulation. The idea behind is that only a class of items needs to be set up (e.g. a certain *Actuator*) and then multiple instances (e.g. 6 identical RCS thrusters) can be mounted (see Section 4.4.1) based on a single template. Thus, only different components have to be defined separately.

In this chapter, working with the *Modelling* tree and a basic description of available models is explained. The theoretical background of the models and specific settings are described in the Model Reference. A detailed example of the *Modelling* tree records for a rocket vehicle can be found in the Create the Model section of the Conventional Launcher Tutorial.

In line with the *ASTOS* GUI, if any modification has been made in the *Modelling* tree, the scenario must be saved using the *Taskbar→Save* button. Then other actions (e.g. a simulation or optimization) can be started. It is also possible - although not recommended - to modify scenario model files (in the folder `\model\astos`) manually and load such modifications using the *Application menu→Reload Scenario* button.

Workflow

The modelling task requires the knowledge of the physics behind the scenario. Typically, it is the most time consuming task required for the simulation of a mission. The sequence, in which model templates are added and specified is irrelevant. The GUI arranges models in the following order:

- **Scenario General Settings** are defined in the topmost tree node, see Section 4.1
- *Environment* includes *Celestial Bodies*, *Atmospheres*, *Hydrospheres*, *Winds* and *Magnetic Fields*, see Section 4.2
- *Vehicle Parts & Properties* includes *Actuators*, *Aerodynamics*, *Aerothermodynamics*, *Components*, *Sensors & Transmitters*, *Power* and *Data*, see Section 4.3

- *Vehicle & POIs Definition* is used to build up vehicles, satellite constellations, points of interest and catalogues, see Section 4.4
- *Dynamics Configuration* is used to set the dynamics of each vehicle and its parts (e.g. actuators) for (optionally) multiple phases, see Section 4.5

To add a model to the scenario, several options exist. This is described in the following using a placeholder [Model] for all available model components (e.g. celestial bodies, vehicles, components, phases, etc.), although the dialog may have different options for specific model components.

1. Select the corresponding button in the *Add* ribbon. Then specify an arbitrary *Identifier* in the *Add [Model]* window and press **Create**.
2. If a model of the desired type already exists in a branch of the *Modelling* tree, it is possible to clone the model by right-clicking on the existing entry and selecting *Clone*. Then right-click the cloned model and press *Rename* to redefine the name of the new model.

Note: For many models, it is necessary to further choose a model *Type* and *Subtype* in the *Add [Model]* window.

Every model is defined with a unique identifier string which can be changed in the *Modelling* tree. A valid string includes letters, numbers and underscores, but cannot start with a number, e.g. *1st_stage* is not valid.

4.1 Scenario General Settings

The scenario **general settings** are accessible by selecting the root node of the *Modelling* tree whose name is identical with the name of the scenario. The following sub-settings are available:

- *Description* is an arbitrary string describing the mission which appears in also displayed in summaries and reports.

  Note: Special characters (e.g. ? or #) are not supported.

- *Active features* lists the *ASTOS* features available within the local ASTOS installation. In case a desired feature is not present in this list, please contact us at service@astos.de.
  The *ASTOS* software can be used for different scenarios and applications (use cases), e.g. simulation, mission analysis, optimization, flexible dynamics analysis, etc. and each such use case usually features specific settings which are completely independent from settings of other use cases.
  In case that the settings for all supported features by *ASTOS* are active and visible in the GUI, this would make it difficult for the user to track those settings which are relevant of a certain use case. In order to mitigate the complexity of the scenario settings the GUI

provides the user the possibility to deactivate certain software models and thus to hide the corresponding model settings in the GUI.
The scenario features available in the *ASTOS* GUI general settings depend on the installed *ASTOS* features and add-ons. Table Table 4.1 contains an exhaustive list of features which can be present in the scenario general settings and the required corresponding installation features or add-ons. In order that a certain features can be used after it has been installed and then enabled in the scenario *Active features*, a corresponding license feature has to be present in the *ASTOS* license.

Table 4.1: ASTOS features on Microsoft Windows

Feature name in GUI	Installation feature	Add-On
Closed loop	**Closed-Loop Simulation**	X
Data system	**Data System Analysis**	X
Error modelling	**Error Modelling**	X
Launch safety	X	**ASTOS Launch and Reentry Safety**
Multibody dynamics	X	**ASTOS Multibody Dynamics**
Optimization	**Optimization**	X
Power system	**Power System Analysis**	X
Reentry safety	X	**ASTOS Launch and Reentry Safety**
Thermal characteristics	**Thermal Analysis**	X

The use case for *ASTOS* with none of the *Active features Enabled* is simulation for mission analysis. If no additional features or add-ons at all are installed, the *Active features* panel is empty. Below a short explanation is provided for each feature in the panel.

- *Closed loop* enables the user to use an *ASTOS* model in *Simulink* by means of an *ASTOS* S-Function block. If this feature is enabled, the *Optimization* feature is disabled and vice versa. The *ASTOS* S-Function allows that an *ASTOS* model can be used in the *Simulink* environment and allows model parameters to be modified from within this environment. The model parameters in the *ASTOS* GUI which shall be accessible in and modifiable from within *Simulink* need to have their *run-time variable* checkbox selected. Then they are then part of the *ASTOS* model control input vector. Details on how to use the *ASTOS* S-Function are provided in the Simulink Interface Manual book.
- *Data system* allows the inclusion of elements for the data sub-system. Details are provided in the chapter Data System Modelling of the Model Reference book.
- *Error modelling* can only be enabled if the *Closed loop* feature is enabled.
- *Launch safety*
- *Multibody dynamics* activates the multibody dynamics modelling in *ASTOS* by enabling the input panels in the vehicle components and in the dynamics, as well as the multibody dynamics equations of motion. More details for the vehicle components are provided in chapter Chapter 7.

- *Optimization* enables the user to perform optimization tasks within an *ASTOS* scenario. If this feature is enabled in the GUI, the *Closed loop* feature is disabled and vice versa. Enabling the *Optimization* feature activates the *Optimization* tree and the *Optimizable* flag in most of the GUI settings. More details on how to perform optimization tasks and on the related features are provided in chapter Chapter 6 and in the Optimization Theory and Description of Methods book.
- *Power system* enables the modelling of an electrical power system onboard the vehicle. More details are provided in chapter Electrical Power System Modelling of the Model Reference book.
- *Reentry safety*
- *Thermal characteristics* enables the modelling of the thermal system onboard the vehicle. Details are provided in chapter Thermal System Models of the Model Reference book.

Note: In case a certain component needs to be added to a scenario but the corresponding *ASTOS* feature is not *Enabled* in the list of *Active features*, then the following "Warning: the feature is not activated, but it is needed to add a component." is issued by the software. In this case, please *Enabled* the corresponding feature in the list of *Active features*. The list is displayed on the right of the GUI when the scenario name is clicked.

The correctness of the scenario modelling can be checked by the user in the predefined report *Scenario Summary* which contains a list of all the models and the phases within the scenario and which can be created by clicking *Start*→ *Info*→ *Scenario Summary* in the GUI.

4.2 Environment

Description

This branch in the *Modelling* tree allows to model the physical properties of the vehicle *environment* with the following specifications:

- *Default Environment* specifies the baseline settings for the environment used in all phases and for all vehicles unless otherwise specified (see below for its configuration)
- *Celestial Bodies* defines shape, gravity, spin state and ephemerides of a celestial body, see section Celestial Body of the Model Reference
- *Atmospheres* defines atmospheric properties (as density and pressure), especially important for the determination of drag forces, see section Atmosphere Models of the Model Reference
- *Hydrospheres* describes all the natural phenomena connected to the presence of water in the atmosphere, see section Hydrosphere Models of the Model Reference

- *Wind* defines the velocity properties of the atmosphere w.r.t. the ground, see section Wind Models of the Model Reference
- *Magnetic Fields* describes the magnetic field properties of the environment, see section Magnetic Field Models of the Model Reference

Examples for a complete setup of a scenario environment can be found in the tutorials. A complete description of all individual model settings and their background can be found in the Environment Models section of the Model Reference.

Configuration

The configuration of the *Default Environment* is explained below. All other environment settings are explained in the corresponding sections in the Model Reference as given in the section above.

The selection of a *Central body* is mandatory, while the definition of *Atmosphere*, *Wind*, *Hydrosphere* and *Magnetic field* models is optional. These drop-down menus allow the user to select from models previously defined in the respective panels (described in the following paragraphs). For optional fields, *None* is selected as default.

The *Gravitational perturbation* panel allows to select/deselect one or more already defined *Celestial Bodies* from a table by a left-click on the respective checkboxes in the *Select* column. The selected bodies will be considered for the computation of the *Third body perturbation* and of the vehicles gravity potential energy. As explained in the Model Reference, the perturbing gravity field of third bodies is simplified and does not consider the *Type* specified in the corresponding celestial body configuration panel, but only its *Gravity parameter* is used. If no selection is made, only the gravity of the *Central body* is considered in the propagation.

In a similar way the *Solar radiation pressure* (see Solar Radiation Pressure chapter of the Model Reference book) and the *Relativistic effects* (see Relativistic Effects chapter of the Model Reference book) settings affect the dynamics of the scenario. They allow to compute the additional accelerations due to solar radiation pressure and due to relativistic effects respectively for the selected *Celestial Bodies*. For the *Solar radiation pressure* the user can decide to use the *General Relativity Formulation* or a *Custom* one. The latter requires that the user provides the values for each coefficient (see Relativistic Effects chapter of the Model Reference book for details).

Background

Celestial body, atmosphere and wind models generally have a direct effect on the trajectory propagation of a vehicle. Magnetic field models only affect the rotational dynamics. Hydrosphere models are used only in the link budget and navigation analyses (for more details refer to Chapter 5) and have no effect on the vehicle dynamics.

Note: Only "active" models, i.e. those selected in the *Dynamics Configuration* affect the trajectory propagation of a vehicle, while only models selected in the *Analyses* configuration panels affect the results of an analysis.

All model types are generally "independent", i.e. a hydrosphere model can be used without having specified an atmosphere model. However certain combinations of model types have to be present for certain analyses.

4.3 Vehicle Parts and Properties

Description

The *Vehicle Parts & Properties* node of the *Modelling* tree defines and lists all models to be used in vehicles and ground stations. Each model can be used in multiple vehicles. These model groups are available:

- *Actuators* contains a list of all defined actuator models, see Section 4.3.1
- *Aerodynamics* contains a list of all defined aerodynamic models, see Section 4.3.2
- *Aerothermodynamics* contains a list of all defined aerothermodynamic models, see Section 4.3.3
- *Components* contains a list of all structural component models like vehicle stages, fairings etc, see Section 4.3.4
- *Sensors & Transmitters* contains a list of all sensor models including transmitters, receivers and cameras, see Section 4.3.5
- *Power* contains a list of all electrical power generation and power management component models, see Section 4.3.6
- *Thermal* contains the a list of all purely thermal models, see Section 4.3.6
- *Data* contains a list of all data storage component models, see Section 4.3.8

Note: Adding a vehicle part model in this branch does not automatically add it to a vehicle!

Note: In case a model is deleted, please check the corresponding configuration in all phases for all vehicles!

Tip: All building blocks defined here serve as templates, i.e. it is sufficient to define identical models (e.g. engines) only once! Multiple instances of these parts are derived from these templates using the *Vehicle Structure* (Section 4.4.1).

Configuration

Add a model by selecting the corresponding button in the ribbon *Add*→ *Vehicle Models* (e.g. the **Actuator** button). Then a window appears displaying the following options:

- *Identifier* expects a unique name (i.e. identifier) for the model
- *Type* selects the kind of the model from a drop-down list

- *Subtype* is only available for few *Type* and selects the subtype of the model

4.3.1 Actuators

The *Actuators* branch of the *Modelling* tree lists all rocket engines and other types of actuators defined in the scenario. These actuators can then be used in multiple vehicles. All actuators listed here serve as templates which can then be added to the vehicle by using the vehicle builder, see Section 4.4.1. More information about the available actuator models is given in section Propulsion Models in Model Reference.

4.3.2 Aerodynamics

The *Aerodynamics* branch of the *Modelling* tree lists all aerodynamics models describing the vehicle aerodynamics (forces and torques) in different configurations during the mission (e.g. launch or re-entry). The number of aerodynamics models is not limited, but in each phase only one model can be selected. More information about the available aerodynamics models is given in section Aerodynamics Models in Model Reference.

4.3.3 Aerothermodynamics

The *Aerothermodynamics* branch of the *Modelling* tree lists all aerothermodynamics models defined in the scenario. They are used to evaluate the aerothermodynamic properties of a vehicle, for example the heat flux in the stagnation point. There is no limitation on the total number of defined models, but only one can be specified for a vehicle in every phase. More information about the available aerothermodynamics models is given in section Aerothermodynamics Models in Model Reference.

4.3.4 Components

The *Components* branch of the *Modelling* tree lists the structural components, the stages and the payload carried by the vehicle. These components can then be used in multiple vehicles. All components listed here serve as templates which can then be added to the vehicle by using the *Vehicle Structure*, see Section 4.4.1. More information about the available component models is given in section Component Models in Model Reference.

4.3.5 Sensors

The *Sensors & Transmitters* branch of the *Modelling* tree lists all sensor models defined in the scenario. Sensors are "passive" equipments, since they do not affect the trajectory propagation but can just provide additional output. They should be used in case the user is interested in evaluating mission analysis aspects such as visibility, link budget, coverage, navigation, etc. They can be used in a simulation, to provide relative position (visibility) w.r.t. user-defined targets and/or for more advanced analyses. All sensor models listed here can be used in multiple vehicles and serve as templates which can then be added to the vehicle by using the vehicle builder, see Section 4.4.1. More information about the available sensor models is given in section Sensor Models in Model Reference.

4.3.6 Power

The *Power* branch of the *Modelling* tree lists all electrical power generation and management components defined in the scenario. These power components can then be used in multiple vehicles. All power components listed here serve as templates which can then be added to the vehicle by using the vehicle builder, see Section 4.4.1. More information about the available electrical power models is given in section Electrical Power Models in Model Reference.

Note: At least one power control and distribution unit (PCDU) is required in a vehicle in order to build working electrical systems! At least one battery needs to be connected to every PCDU!

4.3.7 Thermal

The *Thermal* branch of the *Modelling* tree lists all purely thermal components defined in the scenario. These thermal components can then be used in multiple vehicles. All thermal components listed here serve as templates which can then be added to the vehicle by using the vehicle builder, see Section 4.4.1. More information about the available thermal models is given in section Thermal Characteristics in the Model Reference book.

4.3.8 Data

The *Data* branch of the *Modelling* tree lists all data storage models. These data components can then be used in multiple vehicles. All data components listed here serve as templates which can then be added to the vehicle by using the vehicle builder, see Section 4.4.1. More information about the available data storage models is given in section Data System in Model Reference.

Note: At least one data storage model and at least one transmitter is required to build working data bus structures!

4.4 Vehicles and POIs Definition

Distinct objects like vehicles or ground stations are an essential part of the *ASTOS* modelling. Such model objects are located in the *Vehicles & and POIs Definition* branch of the *Modelling* tree.

To add a new object select the respective button in the ribbon *Add→ Vehicles & Other Entities*. The following objects are available:

- Vehicle
- Ground Station
- Constellation
- Catalog
- Point Of Interest (POI)
- Area Of Interest (AOI)

Each selection adds a new instance of the corresponding object to the *Vehicles & POIs Definition* branch. To configure an object, click on the related tree element. The configuration options depend on the object type and are explained in the next chapters.

4.4.1 The Vehicle Builder: Vehicles, Assemblies, Elements

The **Vehicle Builder** (Fig. 4.1) panel allows the user to build up vehicles and ground stations. The panel is displayed when an object is selected in the *Modelling* tree. The **Vehicle Builder** panel consists of three main sections. The first section (1) shows information about the object name and type, as well as the storage location of the configuration.

Tip: If the *ASTOS* closes without a notification when the *Vehicle Builder* or the *Astroview* panel is opened, please refer to the Troubleshooting chapter, sub-section **GUI closes without notification** in the Software Installation Guide book for a solution.

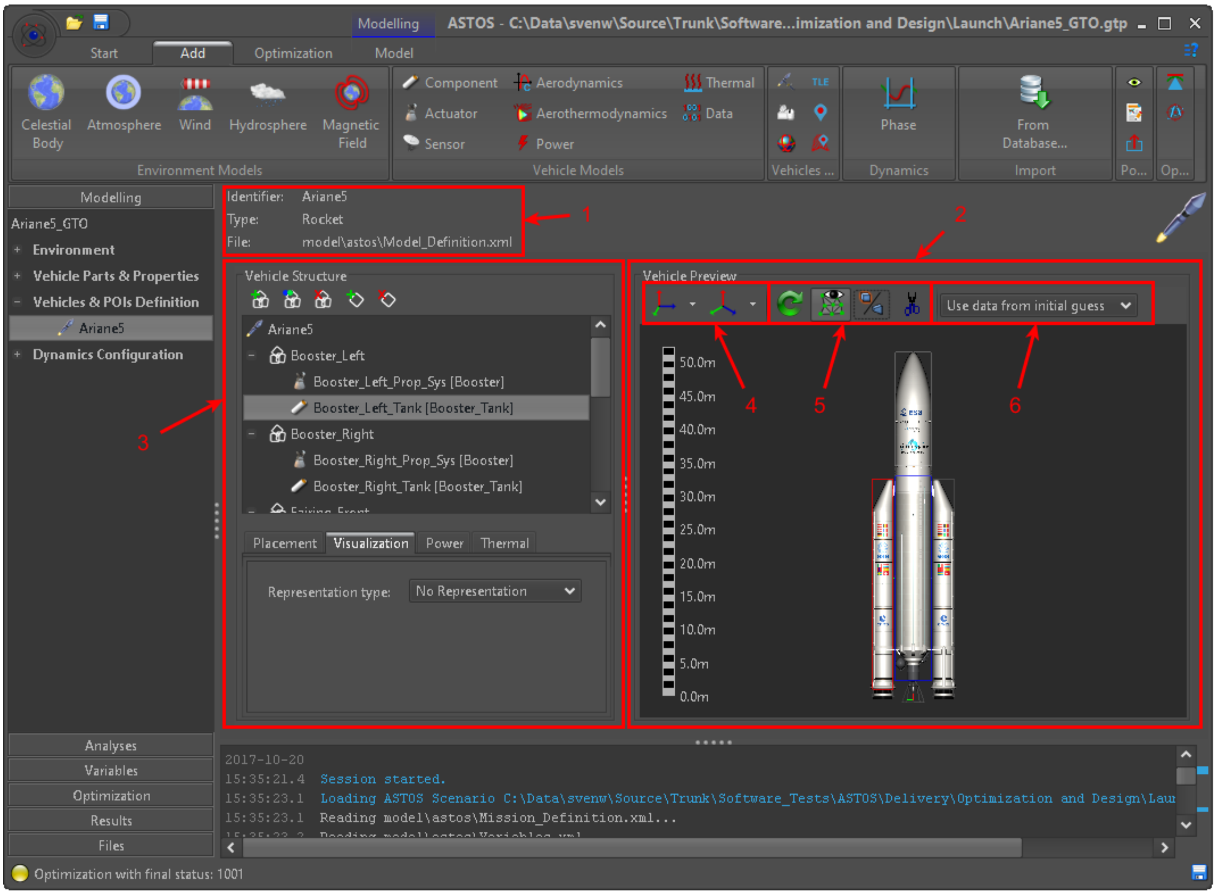

Fig. 4.1: The Vehicle Builder panel

Vehicle preview

The second section (2) contains the *Vehicle Preview*. In the vehicle preview, a 3D model of the vehicle or ground station can be inspected. To **rotate** the 3D model left-click on the vehicle preview and drag the mouse in the desired direction. Similarly it is possible to **pan** the model in the current plane using the right mouse button. **Zooming** is applied by using the third mouse button and by dragging the mouse up and down (or by using the mouse wheel after a left-click on preview panel).

The *Vehicle Preview* also features a **toolbar** which is split into buttons of three categories.

The first category (4) deals with predefined **viewing directions** of two types. The first type includes basic viewing directions (e.g. top, bottom, left) while the second group covers isometric viewing directions (e.g. top-left). One button for each viewing direction is present in the toolbar. The coordinate systems shown on these buttons (and by default) display the last selected viewing direction of each type. The buttons which open a drop-down menu () show all available viewing directions of the corresponding type. After a viewing direction has been selected, the vehicle preview is updated accordingly and also scaled to fit the vehicle or ground station optimally in the viewing area.

The next button category (5) handles **rendering options**:

- This button triggers a reload of the 3D model.
- This button enables/disables the visualisation of 3D meshes. For each element of the vehicle or ground station, a separate 3D mesh can be used in order to define its geometry. In case the visualisation of 3D meshes is disabled, only the bounding box for each element is shown.
- This button toggles between perspective and orthographic projection. In case the orthographic projection is selected, a scale is additionally displayed on the left side of the vehicle preview which gives an indication of the vehicle or ground station size.
- This button enables/disables the cutaway view of the 3D model. Pressing the button opens the *Cutaway Plane Definition* window. Here, the cutaway view can be enabled/ disabled by checking/unchecking the checkbox on top of the dialog. Additionally, the location of the cutting plane needs to be defined by its position and orientation. The orientation is defined by three angles and a rotation sequence. In case the cutaway view is enabled, only those parts of the 3D model are visible, which are located behind the cutting plane.

The last category (6) defines the **data source** from which the bounding box sizes are derived. In case *Use data from initial guess* is selected, the dimensions are directly taken from data specified in the model which is also used to generate the initial guess. If *Use data from simulation* is selected, the dimensions of the bounding boxes are read from the `simulation.struct` file. In this case, the data is taken from the initial guess, a simulation or an optimization.

Creating an assembly

Section (3) is the *Vehicle Structure* and it is dedicated to the vehicle or ground station configuration. Here, the vehicle or ground station is built up from elements, which are defined in the *Vehicle Parts & Properties* branch of the *Modelling* tree. Each vehicle or ground station consists of one or more assemblies, each assembly contains one or more elements respectively.

To add a new element to the vehicle/ground station, first an assembly must be selected from the tree in the *Vehicle Structure* panel. If no assembly is present, a new assembly must be created using the **Add Assembly** button ( ) in the toolbar. This opens the *Add Assembly* window in which the *Assembly type* and an *Assembly identifier* have to be specified. The available entries for the assembly type depends on the vehicle or ground station type.
The button **Clone Selected Assembly** ( ) shall be used in oder to create a clone copy of an already available assembly in the *Vehicle Structure* tree. This will create a new assembly containing identical elements like those in the original assembly. The names of the assembly clone copy and of its elements are generated by augmenting the string *_CLONE* to the names of the original assembly and its elements. The elements in the assembly clone copy need to be configured again because of the newly available element names.
Selecting an existing assembly enables the **Add Element** button ( ), which can now be used to add a new element to the assembly. A click on this button opens the *Add Element* window. Here, all possible element templates (as defined in the *Vehicle Parts & Properties* branch of the *Modelling* tree) are listed and grouped according to the respective categories, e.g. components

or sensors. To include a new element, a template has to be selected from the list and the element name needs to be defined (one element template can have multiple instances in an assembly). A click on the **Add Element** button adds the new element to the currently selected assembly.

Note: Both assemblies and elements can be removed from a vehicle/ground station using the buttons and . There is no **Undo** button for this action.

Configuration of Assemblies and Elements

Below the *Vehicle Structure* tree, the currently selected element can be further configured. All available parameters are grouped into categories, each represented by a tab. The number of available tabs is dependent on the type of the currently selected object. The following categories are available.

Frame definitions: Some coordinate frames can be defined here. It is only available if a vehicle is selected (the top most node). Up to now, only the *Navigation frame* is user customizable (refer to Inertial Navigation Frame in Model Reference).

Placement: The location and orientation of an element is specified here. The placement is always defined relative to another element (i.e. all vehicle parts except aerodynamics and aerothermodynamics models) called the *Reference element* or w.r.t. *Global*, the vehicle's or ground station's **Global Node**.

The **Global Node** of a vehicle or a ground station is the origin with respect to which a vehicle is built up. In the *Vehicle Preview* this is the center of the coordinate system which is displayed when the vehicle or ground station parent node is selected in the *Vehicle Structure*. In some cases the meshes have to be deactivated using the button in order that the global coordinate system is visible.

In case the selected element shall be defined relative to another element, the desired reference element identifier can be selected from the same drop-down menu. Only elements already present in the vehicle structure can be used as reference elements. Additionally, it is ensured that no circular dependencies are created by only providing a subset of the available elements. The connection between the reference element and the currently selected element is defined through the *Reference node* (node of the reference element) and the *Anchor node* (node of the currently selected element), respectively. In case the reference element is set to *Global*, only the anchor node is available. The following nodes are available: *Center*, *Left (y)*, *Right (-y)*, *Bottom (-x)*, *Top (x)*, *Front (z)*, and *Back (-z)*.
The location and orientation of an element can further be modified by the two nominal translations i.e. *Pre-rotation position offset* and *Post-rotation position offset* (location modification) and by the nominal rotation *Nominal orientation* (orientation modification). Nominal means that the provided values are time-invariant and do not change dynamically. Thus, they can be understood as fixed offsets, for example for mounting purpose. A *Pre-rotation position offset* separates the *Reference node* and the nominal rotation axis, whereas a *Post-rotation position offset* separates the nominal rotation axis and the *Anchor node*. First, the pre-rotation position offset is applied to the element, defined in the frame of the reference element.

Then the rotation defined by three *Euler angles* and a *Rotation sequence* is applied to define the orientation and nominal frame of the currently selected element. Finally the post-rotation position offset is applied, defined in the nominal frame of the currently selected element. In case no position offset is defined, the anchor node of the selected element coincides with the reference node of the reference element.

The translation and rotation described above correspond to a nominal, fixed orientation of one element with respect to another. *ASTOS* also allows analogous time-variant orientations of components over time which are independently adjustable for all specified phases. This functionality can be activated by setting respective *Degrees of freedom* to *enabled* (*disabled* by default). Specific settings and ranges for the selected degree of freedom can then be applied in *Dynamics Configuration -> Vehicles & POIs Dynamics* for the selected element. For more details and some examples on that topic refer to Section 4.5.2.4.

Visualization: This category contains all parameters required for the visualization of the selected object. Here, the *Representation type* of the selected object is defined. The following representations are available:

- *No Representation*: No 3D mesh is used for the element visualization, i.e. only the bounding box is visible in the vehicle preview.
- *Automatic Generation*: An automatically generated 3D mesh is be used for the element visualization. The 3D mesh is created or updated each time a simulation or optimization is performed.
- *User Defined*: Import a user-defined 3D mesh created with an external tool (e.g. *Blender*) or with the internal *CAD Import and Texturing Tool*. For this option, name and location of the *Mesh file* must be specified. Supported file formats are *Wavefront OBJ* (`*.obj` with triangular faces) and *3DS Max* (`*.3ds`). Furthermore, a *Mesh scaling* and *X/Y/Z offsets* can be specified to adapt the mesh position and size. It is possible to convert CAD data stored in a STEP file into a *Wavefront OBJ* file using the *CAD Import and Texturing Tool*. For a detailed description of this tool, see Section 4.4.2). The setting for *Blender* are provided in Section 10.1.3.

Power: This category is only available for vehicle elements. Each vehicle can contain one or more Power Control and Distribution Units (PCDU). The power settings differ slightly depending on the selected element type. In case the selected element is a power source, only the *PCDU* needs to be selected from the drop-down list. For all elements which are power consumers, the relevant power circuit needs to be selected additionally from the *Power circuit* drop-down list.

Data: This category is only available for vehicle elements. In case the selected element is a data producer, the attached storage devices needs to be selected in this category. If the element is a storage device, the transmitter needs to be selected which is used to transmit the stored data.

Thermal: This category is only available for vehicle elements. Here, thermal connections between elements can be defined. A thermal connection is defined by the two participating elements, a set of node indices for each element and the thermal resistance of the connection. In general, vehicle elements only provide a single node. Therefore, only the node indices of the

destination element are visible. Normally, the indices are greyed out. Only in case the vehicle element is a *surface element*, the indices can be set.
Defining a connection for one element will automatically define the connection in the settings of the target element. The unit of the thermal resistance is considered global for the current vehicle. Changing the unit for one element will also change the unit for all other elements. Available thermal resistance values are automatically converted to the new unit.

Propulsion: Some additional settings are available for elements derived from *actuator* templates. Apart from defining multiple instances of an identical actuator and locating the individual elements on the vehicle, there is a simplified setup available which can be used when the exact location of an actuator cluster is irrelevant (e.g. for a 3-DOF simulation or optimization). In this case, a *Virtual Engine Multiplier* > 1 can be used to quickly realize a set of identical actuators.

Note: *Virtual Engine Multiplier* has an effect on the multiplication of the total engine thrust, total engine mass flow and total engine mass and shall be used for a simplified setup which does not take into account the effect of the additional virtual engines on the vehicle **CoM** and **Inertia**. Hence *Virtual Engine Multiplier* shall not be utilized for 6-DoF simulation, where the correct **CoM** and **Inertia** need to be used.

Another important property for actuators is that propulsion models (engines) have to be mapped to an *associated tank*. This allows *ASTOS* to determine the propellant source for the respective engine element.

The element configuration provides further input panels which are only available in case certain *ASTOS* features are enabled, e.g. *Optimization*. Such panels are explained in the documentation of the corresponding feature.

4.4.2 CAD Import and Texturing Tool

ASTOS is able to visualize each scenario using 3D triangle meshes. Triangle meshes can be created automatically by *ASTOS* or they can be imported from *Wavefront OBJ* files. The visualisation tool chain is providing the **CAD Import and Texturing Tool** (Fig. 4.2) which is used to convert triangle meshes from CAD data stored in *STEP* files conforming to AP 214, AP 203 and partially AP 209 to Wavefornt OBJ files. In order to open the *CAD Import and Texturing Tool*, open the **Vehicle Builder** (see Section 4.4.1). Afterwards, select any kind of object from the *Vehicle Structure* tree. On the *Visualization* tab, select *User defined* as *Representation type*. Finally, click on the **Open CAD import and texturing tool** button. This will open one instance of the tool.

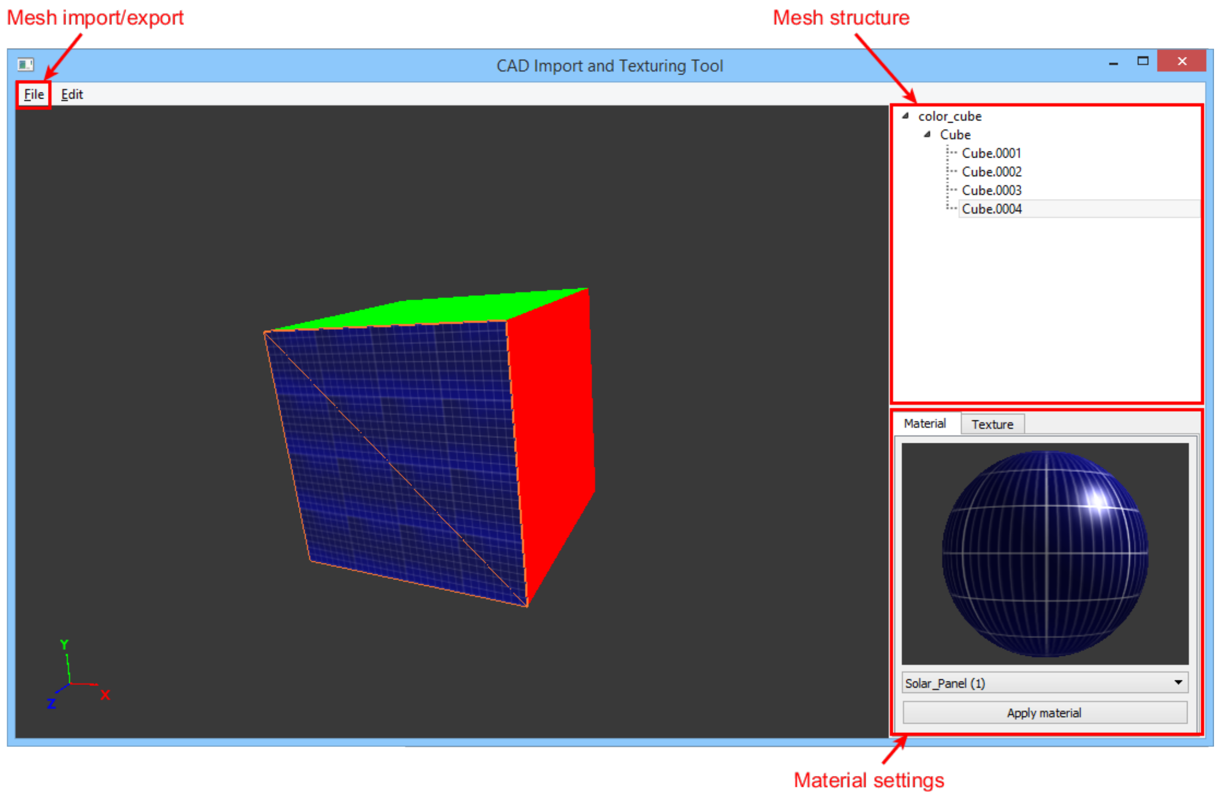

Fig. 4.2: CAD Import and Texturing Tool

Meshes are imported by selecting *File→Open→Wavefront(.obj)* or *File→Open→STEP(.stp/.step)*. In both cases, a file dialog is shown in order to select the file containing the mesh. After selecting the mesh file, a second dialog is shown. This dialog differs slightly depending on the mesh file type. In case the mesh is imported from a *Wavefront OBJ* file, this dialog only provides a selection of the unit used to store the mesh. Meshes exported by the *CAD Import and Texturing Tool* are always stored in meters. In case the mesh is imported from a *STEP* file, this dialog additionally provides input fields for *Linear deflection* and *Angular deflection*. These parameters are required in order to create the triangulation of the shapes defined in the *STEP* file. The structure of the imported mesh is shown by the tree in upper right corner of the main window. The tree contains three levels. The root node identifies the whole mesh. Selecting the root node will select all parts of the mesh. The second level shows the independent submeshes. The third level contains all batches of a submesh. Selecting a node for a submesh will select all batches attached to that submesh. Single batches can be either selected in the tree or by clicking on the mesh itself. Selected batches are marked with an orange outline. After one or more batches have been selected, a new material can be assigned to them. First of all, the desired material needs to be selected from the list of available materials accessible in the lower right corner of the main window. The material list contains predefined materials as well as the materials imported from the mesh file, in case materials are defined. A preview of the material is shown above the material selection. Pressing the **Apply material** button will apply the material to all selected batches. The number of batches currently using a material is written in brackets behind the material name.

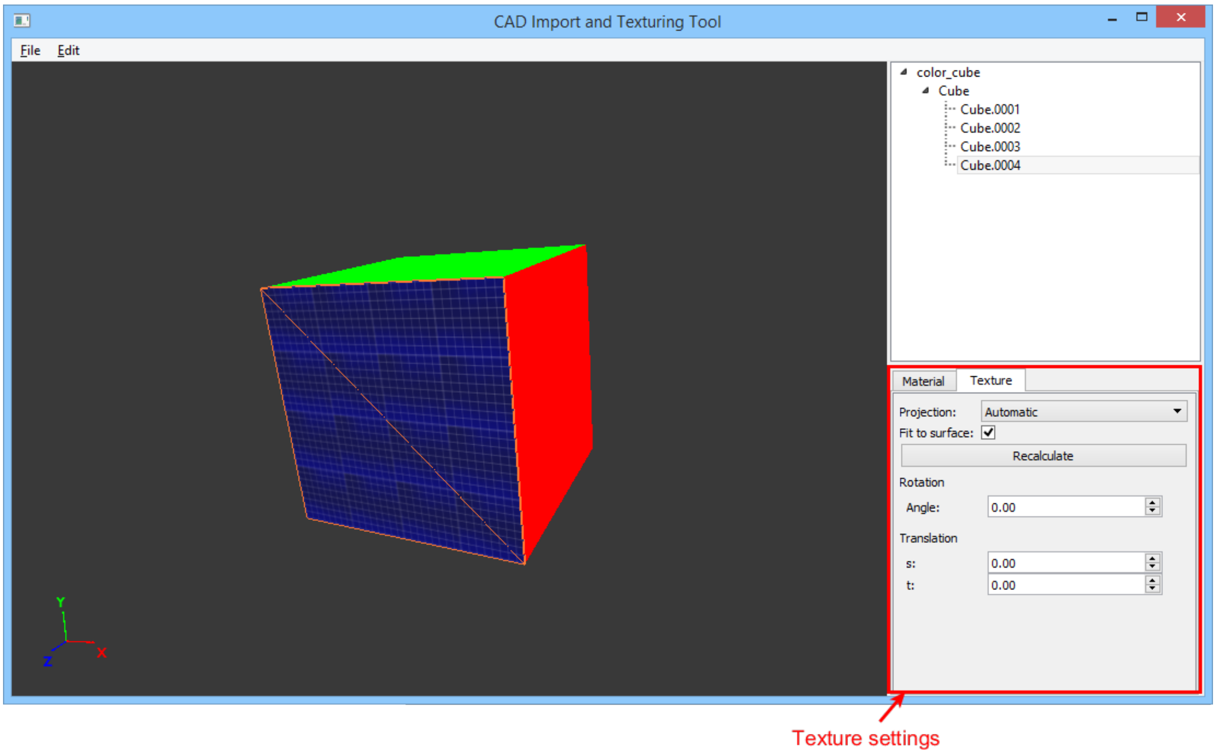

Fig. 4.3: Texture settings

In case the newly assigned material is a predefined material including a texture map, new texture coordinates are computed automatically. Each predefined material owns a physical size. This physical size is considered in the computation of the new texture coordinates in order to map the texture according to the physical size of the selected batches. The automatic mapping already provides a set of high quality texture coordinates but the computation of the texture coordinates can be customized on the *Texture* tab in the lower right corner of the main window. First of all, the *Projection* needs to be specified. Possible values are *Planar*, *Cylindrical*, *Spherical* and *Automatic*. In case *Automatic* is selected, one of the remaining projections is selected based on the three-dimensional extend of the selected batches. The *Fit to surface* parameter enables/disables the optimization of the generated texture coordinates. If enabled, the distortion of the texture introduced by the selected projection is reduced as much as possible. The texture will afterwards follow the surface as much as possible. Straight lines will be bended according to the surface of the selected batches. Finally, the rotation and position of the texture can be customized by setting the rotation *Angle* and the translation in horizontal (*s*) and vertical direction (*t*). All changes are only applied in case the **Recalculate** button is pressed.

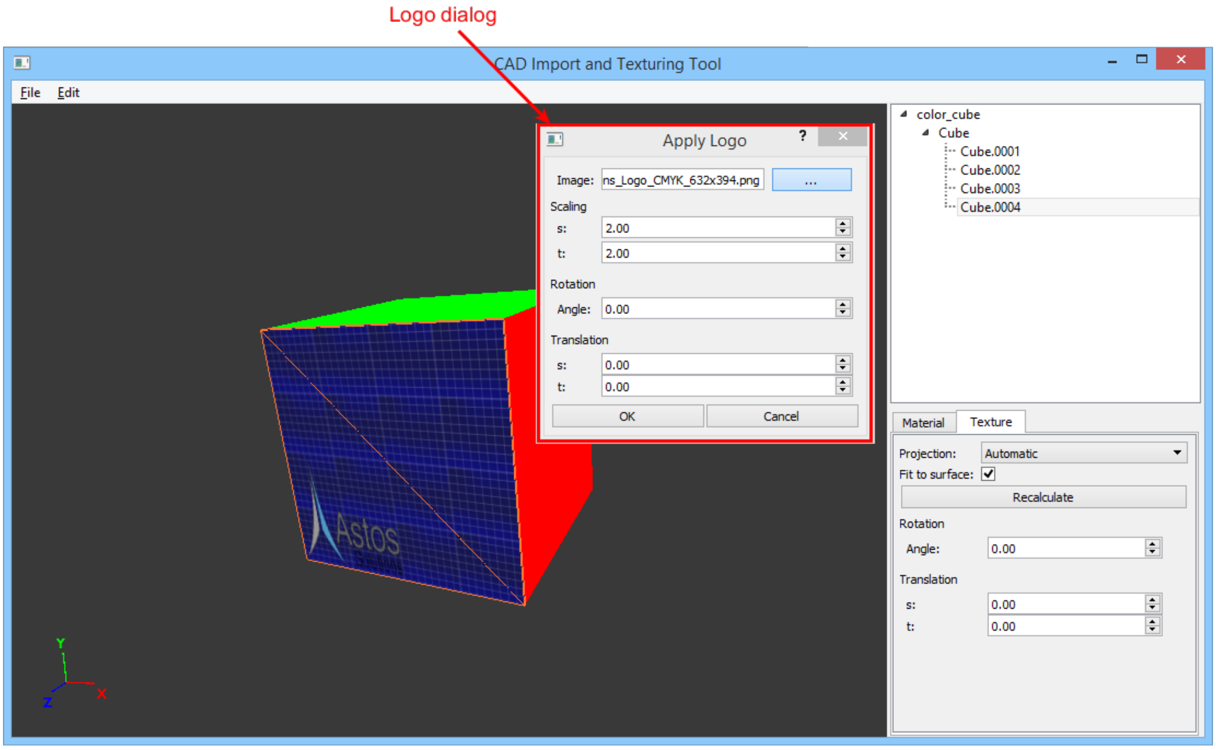

Fig. 4.4: Applying a logo

In case a single batch is selected, a logo can be applied on top of the batch. Selecting *Edit→Apply logo* will open the logo dialog shown in Fig. 4.4. In this dialog, the image file containing the logo as well as the size, position and orientation of the mapped logo can be defined. Pressing the **OK** button will apply the logo to the batch.

4.4.3 Vehicles

The *Type* drop-down menu of the *Add Vehicle* window list the following options:

- *Capsule*
- *Rocket*
- *Satellite*
- *Winged Spacecraft*

The Vehicle type influences the available types of assemblies which can be used in this vehicle. The Vehicle type *Rocket* supports up to 4 rocket stages, boosters, a fairing and a payload. The Vehicle type *Satellite* only supports one assembly type by the name of *Core*.
The Vehicle types and their available assembly types are used to structure the scenario or the vehicle and do not influence the behaviour of the vehicle. In future releases of *ASTOS* the selected Vehicle type or assembly type might also influence the visualization of this vehicle in AstroView. For details on how to structure a vehicle using assemblies and vehicle parts see Section 4.4.1.

4.4.4 Ground Stations

Ground Stations are primarily used to position different kinds of sensors and thereby enabling simulations and analyses of visibilities between vehicles and ground stations or link-budget analyses for data transmission. Like vehicles they are build from assemblies and vehicle parts. Unlike vehicles a Ground Station can not have actuators nor can it jettison one of its assemblies. Since Ground Stations do not move their dynamic behaviour can not be changed. Like any other vehicle a Ground Station requires an *Initial State* in the *Vehicles & POIs Dynamics* node. There only longitude, latitude and altitude are available as position. On how to add a Sensor to the Ground Station see Section 4.4.1.

4.4.5 Points and Areas of Interest

POI

The configuration panel used for points of interest (POI) allows the user to provide the POI *Longitude*, *Latitude* and *Altitude* w.r.t. a selectable *Celestial body* . Therefore, a POI (or AOI) always belongs to the same celestial body.
To provide easy access to a set of points, one or more POI can be grouped in a single object. Each of POI is still handled by *ASTOS* independently, but the set can be accessed as a whole with a common name, e.g. inside a coverage analysis. A new POI is added by providing its *Longitude*, *Latitude* and *Altitude* in the *Coordinates* pane, in the row marked with an * or by creating a new row from the table toolbar. New rows can be created above or below the currently selected row. The currently selected row can be deleted using the corresponding action from the toolbar.

AOI

For areas of interest (AOI), the configuration settings depend on the selected AOI *Type*. In case of a *Polygon* AOI, the configuration is similar to the one for POI, i.e. the AOI is defined as a set of points marking the border of the area. Each point is defined by its *Longitude*, *Latitude* and *Altitude* w.r.t. the *Celestial body* selected. Again, several areas of interest can be grouped in a single object and *ASTOS* treats each area independently. Another area is added by clicking the **Add** button at the bottom of the configuration panel.

Alternatively, an AOI can also represent the shape of a *Country* which can be selected from a drop-down list. In this case, no multiple AOI can be used in a single object. The border of the country is imported from `%ASTOS%\gesop\lib\maps\country_borders.xml`.
The *Latitude Belt* AOI allows the user to define a belt on the central body delimited by two parallels, by specifying the *Minimum latitude* and the *Maximum latitude*. If they are set respectively to -90° and 90°, the AOI covers the whole central body.

If an AOI or a POI is selected for *Target pointing* (e.g. vehicle attitude or component orientation), its centroid (geometric center) is computed and used to determine the pointing direction. In case of a *Latitude belt* AOI, instead of the centroid, the mean latitude and the current longitude of the pointing object are used.

4.4.6 Catalogs

ASTOS provides the possibility to define a set of objects using catalogs. Currently, only Two Line Elements (TLE) catalogs are supported. TLE is a format defined by NASA/NORAD which describes the trajectories for a set of objects. *ASTOS* is not able to access single objects defined inside the TLE catalog, i.e objects are only accessible as a whole. After adding a catalog object, name (and path) of the *Catalog file* containing the TLE data needs to be specified.

4.4.7 Constellations

ASTOS supports the definition of satellite constellations. Before a constellation can be configured, a vehicle of *Type Satellite* needs to be present. A satellite can then be selected as *Template satellite* for the constellation.

Note: Selecting a satellite as template satellite moves the corresponding node from the *Vehicles & and POIs Definition* branch to the lower level branch of the constellation which uses the template. It is neither possible to use this satellite template in another constellation nor to access this satellite independently from other features e.g. from analyses.

Tip: In case a satellite template needs to be used "outside" a constellation as well, clone it before assigning it to the constellation (i.e. choose *Clone* in the right-click menu).

In addition to the template satellite, a list of identifiers must be specified. Each identifier represents one single satellite of the constellation. It is possible to use the **Add Series** in order to add a series of identifiers based on the template satellite name. Each identifier is constructed from the template satellite name adding a hyphen followed by the satellite number. In this case, all the identifiers previously inserted are deleted before adding the new ones.
Again, it is possible to group several constellations within a single constellation object and each constellation is treated independently by *ASTOS*.

4.5 Dynamics Configuration

The dynamic behaviour of an *ASTOS* scenario is set up using the *Dynamics Configuration* in the *Modelling* tree. It is split up into two main categories, *Phases* and *Vehicles & POIs Dynamics.*

Phases

The *Phases* configuration panels are used to create and manage the sequence of flight phases in the scenario, (e.g. vertical ascent, pitch-over, gravity turn, stage separation, etc. for a typical launcher scenario). A definition of multiple phases is necessary whenever discontinuities appear in the scenario schedule. Examples for such events are the jettisoning of components, changing environment or equations of motion, activation of an engine or changing a control law.

The phase management in *ASTOS* also allows to define various conditions for automated phase transitions. Furthermore general simulation settings are defined here.

Vehicles & POIs Dynamics

The *Vehicles & POIs Dynamics* configuration panels are used to define the dynamic behaviour of all existing vehicles in the scenario. This includes the initial state of the vehicle(s) and settings for fundamental vehicle properties (aerodynamics model, active propulsion systems, environment, etc.) throughout all phases in the scenario. Some default settings valid for multiple phases can be defined.

4.5.1 Phases & Common Settings

Description

This chapter describes the phase model of *ASTOS*. A phase is a part of the dynamic of an *ASTOS* scenario in which the modelling is constant, e.g. the equations of motion, the configuration of the vehicle, the environment, etc. In a launcher scenario for example the jettison of stages or the fairing defines the end of one phase. In an interplanetary scenario the switch from one central body (e.g. Earth) to the new central body (e.g. Mars) would require a new phase. The definition of a new phase is required every time one or more of the following settings have to change:

- jettison of an assembly
- activation or deactivation of an actuator
- *Aerodynamics configuration*
- *Aerothermodynamics configuration*
- any part of the *Environment*
- *Equations of motion*
- *Attitude* of the vehicle or of the equipments (e.g. antenna, actuator, etc.)
- *Magnetic moment*
- *Mass distribution*, i.e. inertia and center of mass
- *Flexible dynamics*
- tank *Fueling* and *Ullage pressure*
- *Throttle setting* for engines
- *Link Partner*, *Viewing window* and *Duty cycle* for sensors

Configuration

If *Modelling* tree # *Dynamics Configuration* # *Phases & Common Settings* is selected, the configuration panel displays the following settings:

- *Mission Start Date* specifies the initial date to be used for the simulation, see Section 4.5.1.1

- *Independent variable* defines the kind of the independent variable used for the integration/ propagation of the full mission:
  - *Time (t)*
  - *True Longitude (L)*, the modified equinoctial element L
- *Normalized* defines if the independent variable shall be normalized (*Enabled*), see Section 4.5.1.2
- *Phases*
  This table is used to activate/deactivate phases (*Use* column), merge two phases, split a phase, move up/down or delete a phase, see Section 4.5.1.3
- *Default simulation settings*
  Select integration method, allowed integration error, minimum and maximum step size for the integration and the number of output point per phase, see Section 4.5.1.4.
- *Automatic simulation options*
  Activate or deactivate an automatic simulation action following initialization or optimization and select simulation mode, see Section 4.5.1.4.

Phase List

In the *Modelling* tree → *Dynamics Configuration # Phases & Common Settings* a list of all phases defined in the scenario are listed, also the one not used. Selecting a phase from this list displays its settings in the configuration panel. Since at least one phase is required, in a newly created scenario one phase is present. In order to add a phase, select in the ribbon *Add*→ *Dynamics*→ *Phase*. This opens the *Add Phase* window with the following settings:

- *Identifier*
  Select a unique identifier for this phase.
- *Add after*
  If activated the new phase will be placed after the phase chosen from the drop-down menu. By default, the new phase is appended to the end of the *Phases* list

A right-click on a phase in the *Phases & Common Settings* list opens a menu with the following options:

- *Clone*
  Copy the selected phase with all its settings and add this copy at the end of the phases list.
- *Delete*
  Delete the selected phase.

  > **Note:** There is no **Undo** button! For temporary deactivation of a phase, deselect the *Use*checkbox in the *Phases* panel (see Section 4.5.1.3).

- *Rename*
  Change the identifier of the selected phase.
- *Merge With Next*

The selected phase and the one below are merged to one phase. The settings for *Phase span defined by*, *Independent variable* and *Final time* are taken from the phase below.

- *Split*
  The selected phase is split into two new phases. Both new phases contain the settings from the original.

Phase settings

This chapter describes the options for an individual configuration of each phase. This includes a description and definition of final/duration for the independent variable when the phase should be terminated. On how to set up a phase terminated by values of the independent variable see Section 4.5.1.5. It is also possible to terminate a phase based on events instead of values of the independent variable. This feature is described in Section 4.5.1.6.

4.5.1.1 Mission Start Date

Description

The mission start date defines a fixed point in time for the beginning of the mission. It affects all computations depending on the date, like celestial ephemeris and spin data.

Time standard

ASTOS covers different expressions for the *Time standard* (for general information about time systems, please see [9]):

- *TT*, the Terrestrial Time, is the astronomical standard for the passage of time on the surface of the Earth (TT = TAI + 32.184 s). It uses the second as the fundamental interval.
- *TAI*, the International Atomic Time, is a high-precision atomic time standard which is independent of the average rotation of the Earth (TAI = TT - 32.184 s; TAI = UTC + 37 s, as of July 2015)
- *GPS*, the time of the Global Positioning System, is the International Atomic Time with 19 leap seconds (GPS = TAI - 19 s)
- *UTC*, the Universal Time Coordinated, is the International Atomic Time with leap seconds added to compensate the Earth's slowing rotation. Sometimes the general term Universal Time (UT) is used for UTC (UTC = TAI - 37 s; as of January 2017).

Date Format

Various options are available to define the time format which are related either to Gregorian or Julian dates. They can be specified in the context-sensitive fields which appear after selection from the *Date format* drop-down list.

- *Calendar Date* is the point of time given as Gregorian date with year, month, day, hour, minute and second

- *Time Code A* is a Gregorian date as one line element. The format is *YYYY-MM-DDThh:mm:ss.dZ* with four-digit year *YYYY*, two-digit month *MM*, two-digit day *DD*, fixed character separator *T*, two-digit hour *hh*, two-digit minute *mm*, two digit second *ss*, one-digit decimal fraction of second *d* and fixed character defining the end of the input *Z* (for more details, please see [10]).
- *Time_Code_B* is a Gregorian date as one line element similar to *Time Code A*. The format is *YYYY-DDDThh:mm:ss.dZ* with the only difference that the day of the year is given as three-digit *DDD* with values between *001* and *365* or *366*. For more information about this time code, see [10].
- *Julian Date* is the point of time given as continuous count of days and fractions elapsed since the initial epoch defined as Universal Time January 1, 4713 BC, 12:00
- *Modified Julian Date 2000* is the point of time given as continuous count of days and fractions elapsed since the initial epoch defined as Julian date 2451544.5 TT, or January 1, 2000, 0:00 TT

> **Note:** This is not the epoch J2000.0 which is January 1, 2000, 12:00 TT.

- *Modified Julian Date 1950* is the point of time given as continuous count of days and fractions elapsed since the initial epoch defined as Julian date 2433282.5 TT, or January 1, 1950, 0:00 TT
- *Modified Julian Date* (MJD) is the generic modified Julian date introduced by the Smithsonian Astrophysical Observatory in 1957 because of the large numbers of the original Julian Date. The epoch of MJD is November 17, 1858, 0:00 (MJD = JD - 2,400,000.5).

Offset

Offset adds an offset between *Mission start date* and the beginning of the first phase. To activate it, click on the *default* button. If the *Offset* is set to *custom* the following changes can be made:

- *Offset* specifies a time offset to the mission start date
- *Bounds* are optional, when activating the checkbox next to the label, a lower bound and an upper bound for the time offset can be defined

> **Tip:** By activating the bounds for the offset, the mission start date (e.g. launch) can be made optimizable!

4.5.1.2 Independent Variable and Normalization

Selecting the independent variable

For most scenarios *Time(t)* best suits as an independent variable. On the other hand in case of an orbit transfer scenario, a low-thrust scenario or an interplanetary scenario, it might be beneficial to use the equinoctial element *True Longitude (L)* as an independent variable since the integration variable is an angle rather than time. See Equinoctial Orbital Elements in Model Reference for more details on the equinoctial element.

Note: If *True Longitude (L)* is used, the equation of motion of the vehicle have to be set to *Equinoctial Elements* in all phases. See Section 4.5.2.2 for setting the equation of motion for a vehicle.

Note: If *True Longitude (L)* is used, there may be only one vehicle in the scenario. If the user tries to introduce another vehicle, an error message is shown. On the other hand, if more than one vehicles are present in a scenario, it is not possible to switch from *Time (t)* to *True Longitude (L)*.

Normalized Independent Variable

If *Normalized* is set to *Enabled* the *Independent variable* is normalized throughout all phases of the scenario. This means that in phase 1 the *Independent variable* increases monotonically in the interval [0.0,1.0), in phase 2 the *Independent variable* increases monotonically in the interval [1.0, 2.0) and so on. The default *Normalized* setting is *Disabled*. The advantage of a normalized independent variable is that an optimization of the phase end times (i.e. the unnormalized independent variable) gets more efficient under certain circumstances. Such circumstances are for example very long and very short phases. It is recommended to use normalization if a short phase follows a long phase and the phase times are not fixed (i.e. they are optimizable).

Note: This setting is only available if the **Optimization** feature is active.

The phase initial/duration/final values are stored in three additional real parameters which are automatically created (see Section 6.8) together with one nonlinear parameter constraint *Normalized_Time* (see Automatic Model Constraints in Model Reference).

Switching between Enabled and Disabled Normalization

It is possible to switch the *Normalized* setting between *Enable* and *Disable* or vice versa between two actions. Then, when the next action (simulation or optimization) is run, the *Structure Mapping* window appears and shows the modification which are automatically performed by *ASTOS*. For more information, please refer to Section 2.7.2.

4.5.1.3 Table of Phases

The *Modelling* tree → *Phases & Common Settings* → *Dynamics Configuration* → *Phases* table gives an overview of all phases defined in the scenario. It allows to reorder the phases using the toolbar commands and to activate and deactivate the available phases by a left-click on the *Use* checkbox of each phase. A left-click on the *Use* checkbox of an active phase causes that this phase and all following active phases are deactivated. The same operation applied on an inactive phase causes that this phase and all previous inactive phases are activated.

Tip: It is not possible to have a constraint specified in a phase which is not in use, please check the *Constraints* list in the *Optimization* tree to deselect unused phases from the *Specified* column.

The toolbar allows the following actions:

- merges the selected phase with the next one
- splits the selected phase into two phases. Both phases inherit the settings from the original phase
- moves up the selected phase by one position
- moves down the selected phase by one position
- deletes the selected phase

Note: There is no **Undo** for a delete operation. Please deselect the *Use* checkbox for a temporary deactivation!

4.5.1.4 Simulation Settings and Options

Default simulation settings

In the *Default simulation settings* section two sub-sections are available. The *Integration* contains the following settings:

- *Integration method*
  Select the ODE solvers for the scenario:
  - *Dormand-Prince 4/5* is a variable step size Runge-Kutta-Fehlberg integrator of fifth order which uses the Dormand and Prince formula.
  - *Dormand-Prince 7/8* is similar to above but of the eighth order.
  - *Runge-Kutta-4* is a fixed step size method of fourth order.
  - *MEBDFDAE* (Modified Extended Backward Differentiation Formulae, Differential Algebraic Equations) is a variable step size integrator to solve stiff initial value problems based on [23].
- *Integration error*
  The maximum tolerated integration error without reducing the step size in variable step-size integrators. The default value of 1.0E-8 is a good compromise between accuracy and execution time of the simulation action.
- *Normalized step size*

It specifies whether the provided *Minimum step size* and *Maximum step size* are normalized (*enabled*) or de-normalized (*disabled*).

- *Minimum step size*
  The smallest step size a variable step-size integrator is allowed to use.
- *Maximum step size*
  The largest step size the integrator will use. A variable step-size integrator will start with this value and reduce the step size if the integration error is bigger than the specified *Integration error*. If *disabled* the maximum step size is computed internally.
- *Ignore the minimum step size and just print a warning if step size becomes smaller than the minimum*
  In case of *disabled* the specified minimum step size can be enforced so that the integrator does not reduce the step size below this value. When *Enabled* the integrator is allowed to reduce the step size below the minimum. If the step size is reduced below this level a warning is displayed in the *Execution log*.

The *Output spacing* section specifies how often the results of the scenario shall be evaluated in every phase. These results provide the data for the plots and the visualization in AstroView. Increasing the number of times the output is evaluated increases the accuracy of the plots at the cost of longer execution times. For the *Dormand-Prince 4/5*, *Dormand-Prince 7/8* and *MEBDFDAE* integration methods the number of evaluation time points can be specified with the *Specified by* drop-down menu:

- *Number of points*
  Explicitly specify the number of evaluation time points to be used in every phase. The interval length between two output nodes is then computed internally based on the beginning and end of the phase.
- *Interval length*
  Interval between two evaluation time points. The unit of the interval is displayed to the right of the input field, i.e. *Second* if the variable is *Time* or *Degree* if the variable is *True Longitude (L)*.

In case *Runge-Kutta-4* is selected only the *Interval length* can be specified.

Automatic simulation options

In case the optimization feature is enabled, the following settings are available in the *Automatic simulation options* section:

- *Simulate after initialize*
  Choose a simulation action which should be automatically executed following an initialization.
- *Simulate after optimize*
  Choose a simulation action which should be automatically executed following an optimization.

For further information about the type of simulations see Section 6.5. If the optimization feature is disabled the simulation is always *Connected Phases*.

4.5.1.5 Phase Settings

This chapter describes the individual settings for a selected phase. For each phase selected from the list in the *Phases* node, the configuration panel provides the following options:

- *Description*
  An optional short sentence or keyword describing the phase.
- *Phase span defined by*
  Depending on the selection, either the final value (*Phase End*) of the independent variable or its duration (*Phase Duration*) needs to be provided. Its unit depends on the selected kind of independent variable. Activate the *Bounds* checkbox to optionally specify a lower and an upper bound for the end of phase value. The bounds are only visible if the **Optimization** feature is *Enabled*.

  Tip: These bounds cannot overlap if the **Optimization** feature is *Enabled* and *Normalized* is *Disabled*.

  Note: In some special cases the transformation from duration to final time creates some warning due to the overlap of bounds. An automatic solution is provided, but the user should check the correctness of it. In the case that the result is not acceptable, it is necessary to switch back to the definition of final phase times.
- *Additional phase end conditions*
  Defines one or more conditions whose fulfillment terminate this phase. If the condition is met (e.g. a certain altitude is reached), the phase is terminated as soon as the lower bound for the *Final Time* or *Phase Duration* is reached if the event occurred earlier than the lower bound or exactly at the time of the event if the event is met between lower and upper bound. If the **Optimization** feature is *Disabled* the phase is terminated at the time of the event. In case more than one event is defined the phase can be terminated if one phase condition or all phase conditions are met. See Section 4.5.1.6 for details. By default this is deactivated.
- *Simulation settings*
  Custom overwrites the *Default simulation settings* described in Section 4.5.1.4.

Specifying an open final independent variable is useful in order to determine the optimal duration of e.g.:

- A phase that is terminated by a trajectory-dependent conditions such as the jettisoning of a fairing once the heat flux is below a certain level.
- A coast arc.
- A phase in which only controllable engines are active.

To change the bounds of the phase times during a set of optimization runs (e.g. if they are running against a bound) the *Optimization* tree → *Phase Overview* should be used. In case the scenario is not intended for optimization, then bounds should be changed in *Dynamics Configuration*.

4.5.1.6 Additional Phase End Conditions

This chapter describes the phase end conditions. With this feature a phase can be terminated based on an event instead of a certain value of the independent variable. These conditions are not evaluated during an optimization run, constraints should be used in that situation.

Activating one or more phase conditions

These conditions can be activated by selecting a phase from the *Phases & Common Settings* list and changing the *Additional phase end conditions:* setting to *Enabled*. Additional phase conditions are added by using the **Add** button in the *Additional phase end conditions* panel and choosing a phase condition from the *Choose condition type* window. A complete list of all available phase conditions can be found in Phase-End Conditions and Optimization Constraints of the Model Reference book. When more than one phase condition is added in the phase settings, it is possible to trigger a phase end if

- One of the conditions is fulfilled or
- All conditions are fulfilled.

The initialize action stops the integration somewhere between the lower bound and the end of phase value as soon as the condition is fulfilled. That means that if the event described with a phase condition occurs earlier than the specified lower bound of the phase time, the phase will be terminated at the lower bound value! See also Section 6.4.3 for additional information on how to use phase conditions.

Tip: If the upper bound value is identical to the end of phase value (respective duration) of the phase time, the integrator will keep these items identical: e.g. if the phase condition stops the phase at 103 second, also the upper bound is set to 103 second.

Note: The *Virtual_ignition_time* and *Virtual_shut_off_time* real parameters are not updated by the phase end condition: they remain set to the final time of the previous and current phase as if the condition is not present. It is possible to manually change the parameters or to let the optimizer change them.

Available settings for phase end conditions

The available settings for a phase end condition depend strongly on the type of the selected condition. The settings common to all phase condition are:

- *Is active*
  disabled to temporarily deactivate a condition. Default is *enabled*.
- *Vehicle ID*
  Select a vehicle from a drop-down list of all defined vehicles in the scenario for which the selected event shall terminate the phase.
- *Phase ends if*
  Select *value equal or greater than reference* or *value equal or smaller than reference* from the drop-down list.

- *Reference*
  Specifies a reference value and a unit for the phase condition.

Many phase conditions do not require additional input. Since the phase conditions are quite similar to the corresponding constraint, additional information can be found in chapter Phase-End Conditions and Optimization Constraints of the Model Reference book.

4.5.2 Vehicles & POIs Dynamics

This chapter and its subchapters describe how to set up the dynamics of vehicles as well as the dynamics of components.

Initial state

An initial state of a **vehicle** defines the position, velocity and attitude at the beginning of a simulation or an optimization. The initial state can be defined in various coordinate systems and representations, e.g. Keplerian orbital elements for satellites or altitude, longitude and latitude for rockets on a launch pad. An initial state also has to be provided for satellite **constellations**, **ground stations** and vehicle **components**.

Default settings

Default settings for a **vehicle**, its **components** and **ground stations** are used throughout all phases. Available options depend on the type of the object, e.g. equations of motion, attitude control strategy or aerodynamics configuration in case of a vehicle.

Settings for individual phases

All *Default Settings* can be adjusted individually for each phase. Additionally, active propulsion systems during a phase and assemblies to be jettisoned at the end of a phase can be selected for vehicles.

Settings for components, actuators and sensors

Similar to the dynamic settings for a vehicle the components, actuators and sensors that belong to the vehicle assembly can also have dynamic, phase-specific settings. If any vehicle part supports for such settings, additional branches are displayed below the node with the vehicle name, e.g. *Vehicles & POI Dynamics -> [Vehicle Name] -> Actuators*), if an engine with thrust vector control or a throttleable engine exists for this vehicle.

4.5.2.1 Initial State

An initial state is a set of start values which are used for the integration executed by the initial guess generator and for the simulation action without previously optimizing. The initial state is not directly used during optimization or during a simulation after an optimization action. In the latter cases, the initial state information is read from a TOPS file.

The *Initial State* tab allows to setup various initial conditions. This tab is present for the following object types with different initial state options (see chapter Initial State Definition in the Model Reference book for a complete description of all individual configuration options):

- **Constellations**
  The initial *State type* for satellite configurations has two different options (see chapter Constellations in the Model Reference book for more details):
  - *TLE catalog* as an option to import initial state data from Two Line Elements
  - *Orbital Planes* as an option to manually configure a constellation as *Walker Constellation* or by defining individual *Custom* orbital planes

- **Ground stations**
  For ground station objects, only a celestial body fixed position needs to be be specified (see chapter Ground Stations in the Model Reference book for more details).
- **Vehicles**
  The *state type* for vehicle objects can be defined by four main categories some of them with various sub-settings (see chapter Vehicles in the Model Reference book for more details):
  - *Position & Velocity* requires inputs in Cartesian or polar coordinates. This option is mainly used for vehicles in the atmosphere or on the ground (e.g. on a launch pad).
  - *Orbital Elements* is an option to quickly set up orbits for various conditions.
  - *Relative Motion* is used to define an initial state relative to another (reference) vehicle.
  - *Barycentric Rotating* allows to set up an orbit in the Circular Restricted Three Body Problem (CR3BP) by specifying cartesian elements for position and velocity in a two-body rotating frame.

These initial states can be represented in various *Coordinate systems* depending on the selected *State type*. Refer to Model Reference for a complete description of all Coordinate Systems.

In addition to the state specific settings, further input can be specified:

- The *Attitude* initial state settings are only available, if *State* is selected in the *Attitude* settings in the tab for the first phase. This setting can be used to define the rotational initial state of a vehicle. Inputs for both attitude and inertial body rates can be specified. (see Model Reference for a complete description of Attitude Definition options).
- The *Epoch differs from mission start date*
  settings allows the user to define an epoch of the vehicle initial state which differs from the mission start date (see Model Reference for information about Epoch, Time Scales and Time Formats). This feature is useful, e.g. if state data is taken from an "older" observation of the vehicle or from catalog data.

- The *Inertial navigation frame* settings are optional and allows the user to define the *Azimuth* of an inertial frame aligned with a launch pad. This setting is only required when inertial navigation control laws are used, otherwise the settings have no effect (see Control Laws and Inertial Navigation Frame in the Model Reference).

4.5.2.2 Default Settings

The *Default Settings* tab allows to specify a number of settings which are common to all phases. Unless explicitly overwritten for individual phases, all phases use default settings for:

Note: *Environment* is the only default setting for ground station objects, all other settings apply to vehicles.

- *Environment*
- *Solar radiation pressure properties*
- *Aerodynamics configuration*
- *Maneuver at beginning of phase*
- *Equations of motion*
- *Attitude*
- *Magnetic Moment*
- *Mass distribution*
- *Flexible Dynamics*
- *Auxiliary states*

Environment

By *Default*, vehicle and ground station objects use the environmental settings defined for the entire scenario as specified in the *Modelling* tree (*Environment -> Default Environment*).

It is also possible to select an *Individual* environment (central body, atmospheric model, etc.) for all phases of a specific vehicle. This enables a respective section in the configuration panel which contains identical settings as for the *Default Environment* of the scenario (see Section 4.2). Unless overwritten again in the settings for individual phases, this environment is used in all phases for this vehicle.

This feature allows scenarios with multiple vehicles and multiple phases where each vehicle has its own environment, e.g. a scenario where one vehicle orbits the Earth and another vehicle orbits the moon in the same phase.

Solar radiation pressure properties

This panel is available only if the *Solar radiation pressure* is enabled for the current *Environment*. It allows to define the vehicle properties relevant for the solar radiation pressure computation (see the Solar radiation pressure section for more details).

Aerodynamics configuration

Select the default aerodynamics model (see Section 4.3.2) to be used for the vehicle. The *Aerodynamics configuration* drop-down list contains the identifiers of all aerodynamics models defined in the *Modelling* tree (*Vehicle Parts & Properties -> Aerodynamics*). By default, no aerodynamics model (- *None* -) is assigned.

Note: In case an aerodynamics model is deleted from the *Aerodynamics* branch, please check the *Aerodynamics configuration* assignment in the respective phases.

Maneuver at beginning of phase

Maneuvers at the beginning of a phase change instantaneously one or multiple states of the vehicle, such as velocity magnitude and/or direction. Optionally, the related propellant consumption required for such a maneuver can be computed. More information is given in section Maneuvers at the Beginning of a Phase in Model Reference.

Note: This feature is not available for the first phase, which is directly defined by the initial state!

Equations of motion

It specifies the representation of the 6 translational states (position vector and velocity vector) and the differential equations of motion used to integrate the trajectory of the vehicle. The drop-down list in the *Equations of motion* (EoM) panel contains several options (see Translational Dynamics in Model Reference).

Attitude

This setting specifies the attitude control strategy for the vehicle, i.e. the vehicle's orientation as a function of time. A complete list and explanation of the available *Attitude* definitions is provided in section Attitude Definition in the Model Reference book. Some of the *Attitude* definitions provide a *Control/State* selector which allows the user to enforce a certain attitude on the vehicle (*Control*) or to enable a 6 degrees of freedom (6-DOF) vehicle dynamics (*State*):

- The *Control* option allows the user to enforce a user-defined time-variant or time-invariant attitude on the vehicle while the effect of any forces and moments on the vehicle attitude is not considered.
- The *State* option activates an attitude dynamics model for the vehicle and the effect of any forces and moments on the vehicle attitude is considered.

Magnetic Moment

Magnetic Moment defines the residual magnetic dipole moment vector of the vehicle. In case an external magnetic field is defined in the scenario it creates a torque on the vehicle. It also interferes with the measurements of the *Magnetometer*. If *enabled* the dipole moments in all three axes of the body frame of the vehicle are required user input.

Mass Distribution

Mass Distribution is an optional setting to define the center of mass and inertia tensor for the vehicle. If *disabled* these are computed automatically. This panel is described in detail in Mass Distribution and Inertia Computation in Model Reference.

Auxiliary states

Auxiliary states are optional auxiliary differential equations which are evaluated during a simulation and *disabled* by default. They can be *enabled* for informative reasons only or to use them as certain cost and constraint functions in an optimization. A complete description is provided in Auxiliary States in Model Reference.

4.5.2.3 Settings for Individual Phases

Apart from default settings for each vehicle, it is also possible to specify individual settings for a single phase of each vehicle. They can be accessed via the respective tabs named according to the phase identifiers. Phase specific settings can include:

- *Aerodynamic configuration*
- *Jettisoned assemblies*
- *Active propulsion systems*
- *Maneuver at beginning of phase*
- *Environment*
- *Solar radiation pressure properties*
- *Equations of motion*
- *Attitude*
- *Flexible dynamics*

Most options are totally equivalent to the ones in the *Default Settings* tab (Section 4.5.2.2). To overwrite these default settings for the selected phase, use the toggle buttons at the top of the configuration panel. The following options exist:

- *Default*
  The settings for the phase are inherited from the *Default Settings* tab.
- *As Previous Phase*
  The settings for the phase are inherited from the preceding phase (not available for the first phase).
- *Individual*
  Redefine the settings for this phase and overwrite the default values.

Tip: All phases which use phase specific settings (i.e. *From previous phase* or *Individual* is selected) are indicated by '°' preceding the tab name.

Jettisoned assemblies

Jettisoned Assemblies allows to jettison vehicle parts, e.g. to model stage separations of a launcher. The related table lists assemblies defined for the current vehicle by their *Assembly ID* (see Section 4.4.1 on how to create assemblies). Select an assembly to be jettisoned by clicking on the desired checkbox in the *Select* column. All vehicle parts that belong to the selected assembly are then jettisoned at the end of the selected phase. These assemblies are then automatically removed from the tables in subsequent phases.

In case a vehicle stage component is jettisoned which feeds an active engine, *ASTOS* automatically introduces a final boundary constraint at the end of this phase, which forces the amount of remaining fuel left to be non-negative. If the **Optimization** feature is available, this constraint can be manually changed to an equality constraint, i.e. the fuel is forced to be burned completely. See Section 6.2 on how to use constraints.

Active actuators

The *Active actuators* table lists all available actuators defined for this vehicle which have not been jettisoned in previous phases. Actuators which shall be active in a phase are selected by clicking on the respective checkbox in the *Select* column.

4.5.2.4 Settings for Components, Actuators and Sensors

In addition to the vehicle itself, also *Actuators*, *Components*, *Sensors & Transmitters*, *Power*, *Data* and *Thermal* models of a vehicle may have a dynamic behaviour. In this case, additional branches are displayed below the node with the vehicle name, e.g. *Vehicles & POI Dynamics -> [Vehicle Name] -> Actuators* in case of an actuator model. The prerequisite is that the model has the respective capability for a dynamic behaviour. This capability can result from:

- Rotational and/or translational degree(s) of freedom (see paragraph in Section 4.4.1)
- Model specific attributes which are set to from *Common* to *Individual* in the *Vehicles & POIs Definition* (e.g. the *Throttle setting* of an engine or the filling ratio of a stage component)
- Model type (e.g. only for transmitters the user can define *Link Partners*)
- Combinations of above

As an exception, the node *Sensors & Transmitters* is always present when a sensor model is mounted on a vehicle. In this case, the option to specify a *Viewing Window* is always provided (see also below).

The general structure of the configuration panel is similar to the configuration panel for a vehicle, i.e. tabs are present for *Default Settings* and for the individual phases of a scenario (see previous sections). The available options depend on the specified dynamic behaviour.

Orientation offsets

If an orientation offset is *enabled* as *Degrees of freedom* for a vehicle part (see Section 4.4.1), the *Rotation [...] around [...] axis* can be specified depending on the *Rotation sequence* selected in the *Vehicle Structure*. For each rotation axis the following control strategies exist and are selectable from the *Control Law* drop-down list:

- *Constant Law*
  The offset with respect to the nominal orientation has a constant value. *Use final value from previous phases (0.0 in case of first phase)* is selected by default, i.e. an offset is inherited from the previous phase. To modify the default setting, deselect the checkbox and provide the desired value.
- *Linear Law*
  The vehicle part rotates with a constant angular velocity about the selected axis with respect to the reference element. *Slope defined by* determines if the angular velocity is set explicitly (*Rate*) or implicitly by selecting a *Final* angular offset. In the latter case, the angular velocity is calculated based on the duration of the current phase (first phase when specified in the *Default settings* tab).
- *Profile Law*
  The orientation varies over time according to a profile defined in a profile data table (see also Section 2.10.2).

Note: These orientation offsets are defined with respect to the nominal orientation of the element.

The nominal orientation of two components is given as shown on the left of Fig. 4.5, i.e. Component2 is rotated 45° about an out-of-plane axis with respect to Component1. Assume that an orientation offset of -90.0° is defined for Component 2 about the same axis for simplicity. The rotation axis is located in the *Anchor node* at the bottom of Component2 (origin of the Component2 coordinate system). The result of the rotation is shown on the right of Fig. 4.5. It is also possible to use rotation axis away from the *Anchor node*, see Example 3 below.

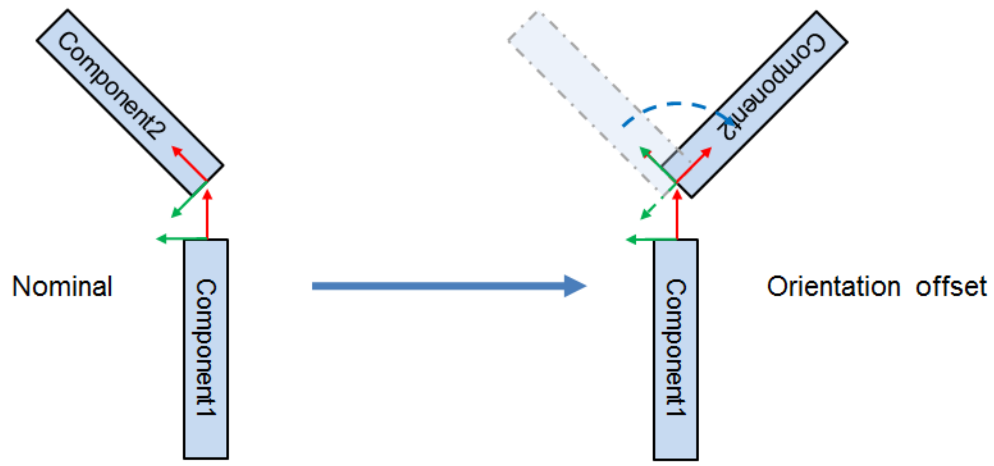

Fig. 4.5: Example for an orientation offset of Component2 in one axis: nominal orientation (left) and after rotation (right)

Position offsets

Two types of dynamic position offsets are available, a *Pre-rotation position offset* and a *Post-rotation position offset*. They need to be *enabled* as *Degrees of freedom* for a vehicle part in order to be available in the dynamics configuration. Both realize a translational motion of the selected element (i.e. vehicle part) with respect to the nominal position. They only differ in the coordinate frame in which the motion is described. A *Pre-rotation position offset* is measured in the frame of the *Reference element* of the currently selected vehicle part, while the *Post-rotation position offset* is measured in the nominal frame of the selected element itself. A *Pre-rotation position offset* separates the *Reference node* and the nominal rotation axis, whereas a *Post-rotation position offset* separates the nominal rotation axis and the *Anchor node*.

The nominal orientation of two components is given as shown on the left of Fig. 4.6, i.e. Component2 is rotated 45° about an out-of-plane axis with respect to Component1. A position offset of Component2 in positive X-direction shall be realized. Depending on the position offset type, different results are obtained. The pre-rotation position moves Component2 along the X-direction of the reference element (Component1), the post-rotation position moves Component2 along the X-direction of Component2.

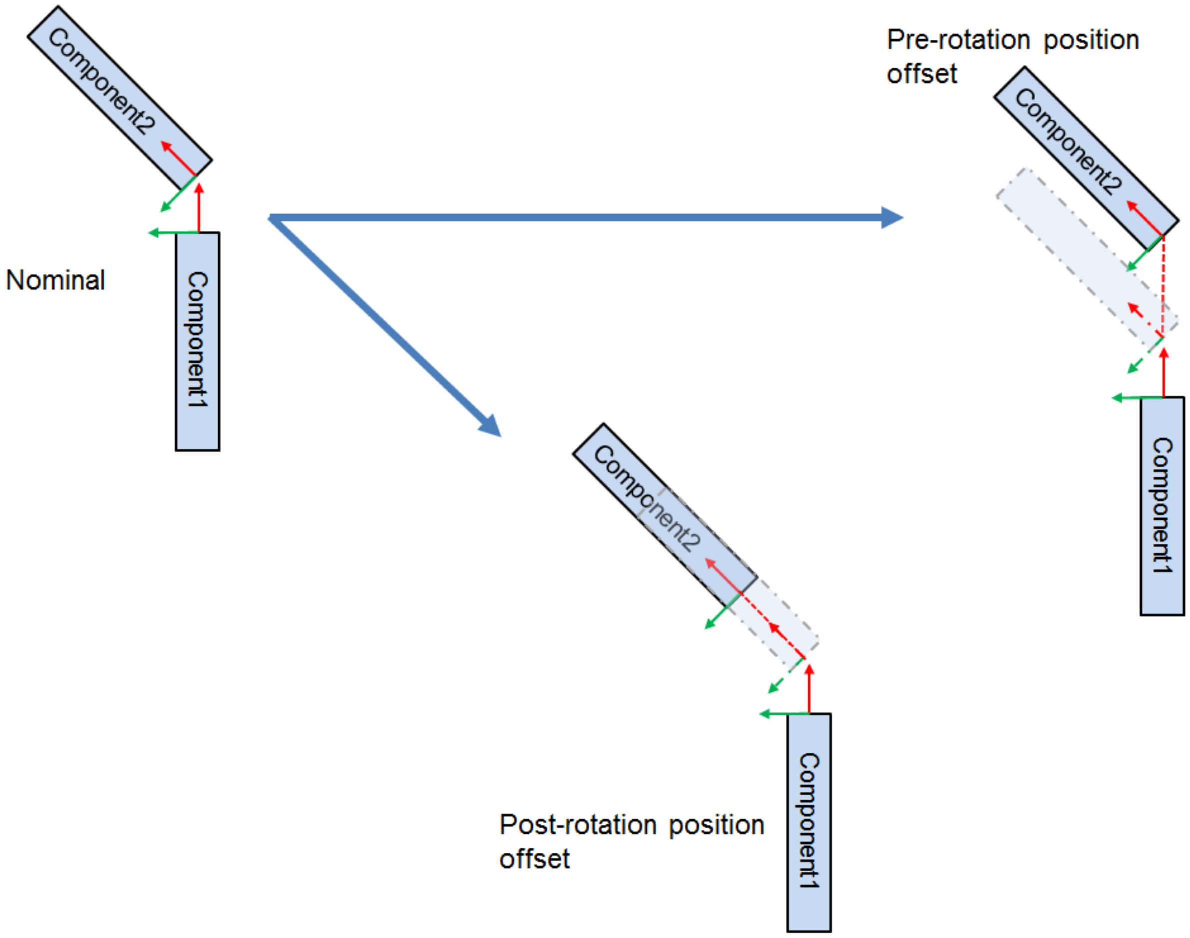

Fig. 4.6: Difference between position offset types: Nominal orientation (left), post-rotation position offset in positive X direction (centre) and pre-rotation position offset in positive X direction (right).

Rotation about an arbitrary point

By default, rotation axes intersect at the *Anchor node* of the selected vehicle part. However, a rotation axis can be moved to an arbitrary location (with respect to the element to be rotated) by combining pre- and post-rotation position offsets.

The nominal orientation of two components is given as shown on the left of Fig. 4.7. Component1 is the *Reference element* with the *Reference node* defined at its top. The *Anchor node* of Component2 (located at its bottom) coincides with the reference node. First, a pre-rotation is applied in positive X-direction and negative Y-direction (w.r.t. the Component1 frame). This separates reference node and anchor node, while the rotation axis still intersect the anchor node (centre of Fig. 4.7). Then, a post-rotation is applied in positive Y-direction (defined in the Component2 frame). This separates the rotation axis and the anchor node (right of Fig. 4.7).

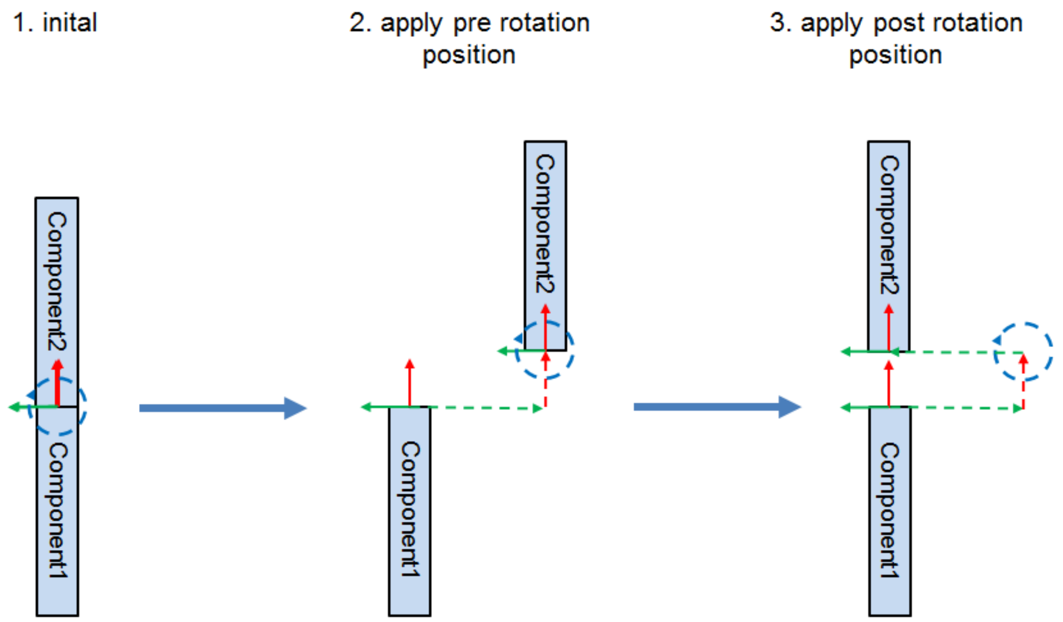

Fig. 4.7: Locating a rotation axis (dashed circular arrow) at an arbitrary point: nominal orientation (left), after pre-rotation offset is applied (centre) and after post-rotation offset is applied (right)

Assume that above situation is realized as "new" nominal orientation in the *Vehicle Structure*. Assume further that a *Linear Law* with *Final* value of -45.0° for a rotation about the Z-Axis is selected in defined as a *Default settings* tab of Component2. This leads to a motion as shown in Fig. 4.8.

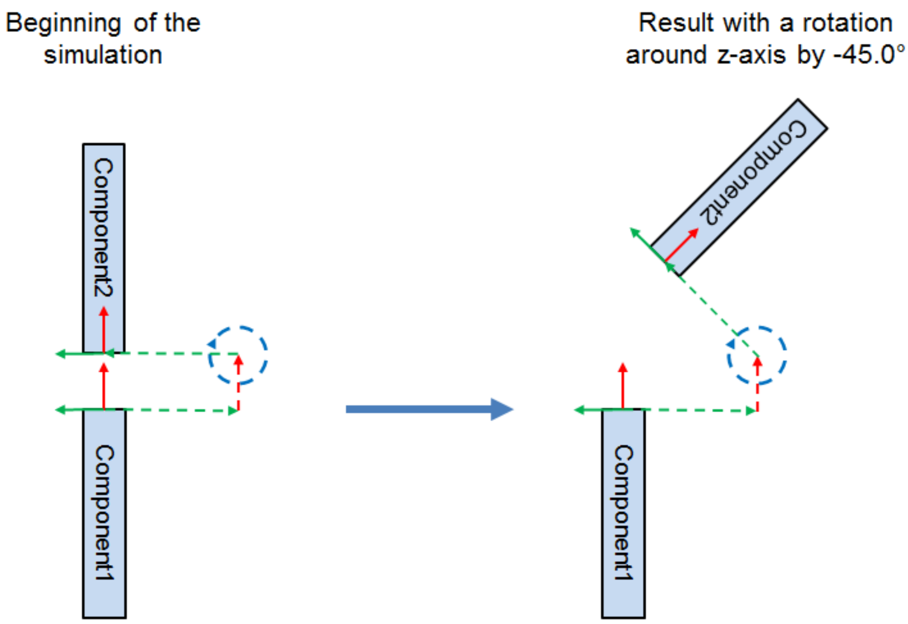

Fig. 4.8: Rotation about the Z-axis with arbitrarily located rotation axis: initial state of simulation (left) and resulting situation at the end of the phase (right).

Link partners for transmitters

This feature is available for all types of transmitters, including transceivers. It allows to select a list of receivers, which are allowed/able to receive the data transmitted by the current object. The table in the configuration panel lists all available receivers (and transceivers) which are defined in the scenario by their *ID*. Desired partner(s) are selected by clicking on the respective *Select* checkboxes.

Viewing window

This feature is available for all types of sensors and allows the user to limit the visibility of the sensor. If the *Viewing Window* is enabled and specified as *Value*, the minimum elevation angle (which is the supplementary angle of the declination) of the field of view is set: whatever is below the minimum elevation angle, it is not seen by the sensor. Sometimes the field of view of a sensor might have an obstacle (e.g. building, tree, mountain, etc.) in a certain direction (azimuth angle in the y-z plane). In this case the user can define this mask by means of a profile data table object (see Section 2.10.2) by selecting *Profile* in the *Defined by* selector.
For more details on how to configure the viewing window, refer to the dynamics configuration paragraph in chapter Sensor Models of the Model Reference.

Duty cycle of Sensors

By default any sensor is always active, e.g. a camera in video mode is always recording data and an antenna may always send or receive data. By enabling the *Duty cycle* of a sensor it is active or inactive based on conditions. The following *Activation conditions* can be defined:

- *Max eclipse*: Deactivate the sensor if the eclipse value is above the given threshold. With a value of 0.0 the sensor is only active in full sunlight and will be deactivated as soon as it enters a partial eclipse. With a value of 1.0 the sensor will remain active even in a full eclipse.

- *Min battery SoC*: Deactivate the sensor if the battery state-of-charge (SoC) falls below the given threshold. In case more than one battery is attached to the same PCDU as this sensor, the lowest state-of-charge of all connected batteries is considered. If electrical systems are not modelled this setting has no influence on the sensor activity status.
- *Data Link*: This option is only available for transmitters and transceivers. If this option is selected the sensor is deactivated if there is no mutual visibility with any of the specified link partners or the link partners are technically not compatible.

As soon as at least one criteria for deactivation is met, the sensor is deactivated.

4.5.2.5 Modelling a Scenario with MBS

Modelling and simulating a scenario with *Multibody dynamics* equations of motion means that the dynamics of each single component with a structural mass is considered as a separate body in the multibody system dynamics.

5 Running Analyses

This chapter gives an overview of the **Analysis** feature of *ASTOS*. An analysis is a form of post-processing of a simulation or optimization. It typically uses a previously created simulation output file (`.struct` file) as data input and generates additional output data such as functions and scalars. This output data is either added to the input simulation file or is written to dedicated results files. Such a two stage simulation approach provides some advantages:

- Complex and slow analyses do not need to be made at every simulation run but only if it is really desired
- Analyses can be performed based on external simulation data
- The implemented analysis concept supports even more complex processing like built-in Monte Carlo analyses

An analysis can be created via the *Add→Post-Processing→Analysis* ribbon button. Then an identifier and the type of the analysis have to be specified. Afterwards the analysis will be listed under the *Analysis* tree node. An overview of available analyses and their configuration is described in chapter Analyses in the Model Reference book.

Analyses further typically provide templates for related reports that serve as a good starting point for customized reports (see Chapter 8 for details on reports). A report template comprises static text as well as placeholders that will be replaced by output form the simulation and analysis runs once the report is generated.

The following terms are required for the explanations further below:

- **Automatic** specifies that the analysis is executed automatically after a simulation run
- **ASOR** (Automatic Simulation Output Refinement) specifies whether additional output nodes shall be added to the selected analysis or not. Analyses are able to refine the output spacing by adding further output nodes in order to give adequate results.

> **Note:** If ASOR is active, output nodes will be only added during simulation runs and not during an analysis run!

Analysis ribbon task

When one of the existing analyses in the *Analyses* tree or the main tree node are selected, the *Analysis* contextual task is shown present in the ribbon. It contains actions which are grouped as follows:

- *All Analysis* ribbon band:
    - *Analyze* runs immediately all analyses
    - *Create Report Templates* creates a default report template for each analysis
- *Selected Analysis* ribbon band actions are only related to the selected analysis of the *Analyses* tree:
    - *Analyze* runs immediately the analysis
    - *Create Report Template* creates a default report template
    - *Automatic* activates/deactivates the automatic computation flag
    - *ASOR* activates/deactivates the ASOR flag
    - *Move up* changes the order of the analyses by moving the selected anylsis one up
    - *Move down* changes the order of the analyses by moving the selected anylsis one down
    - *Rename* renames the analysis
    - *Clone* creates a clone (copy) of the analysis
    - *Delete* deletes/removes the analysis from the scenario

Analyses panel

The analysis panel is displayed when the root node of the *Analyses* tree is selected. It consists of the following elements.

- **Analyses table**
  On the left of the panel a table lists all analyses in the scenario. This table features additional settings that are not available in the individual analyses. Each row of the table contains one analysis.
  The table columns have the following meaning:
    - The first column contains the analysis identifiers
    - The check box in the second column of the table specifies the automatic flag
    - The checkbox in the third column activates/deactivates the *ASOR* flag

  The buttons above the table are used as follows:
    - The arrow buttons are used to change the order of the analyses in the table by moving the selected analysis up or down. The order in the table represents also the order in which the analyses are executed, which might become important if several analyses add data to the same simulation file or in case one analysis is based on the results of another one.
    - The third button is used to delete the selected analysis/analyses.
- **Analyses actions**
  On the right of the Analyses panel are four buttons:

- **Perform All** runs immediately all analyses
- **Perform Selected** runs immediately the selected analysis/analyses of the table
- **Create Report Template For All** creates a default report template for each analysis
- **Create Report Template For Selected** creates a default report template for the selected analysis/analyses of the table

Multiple analyses can be selected while holding the **CTRL**- or **SHIFT**-key and clicking on an analysis of the table.

Note: In the table multiple analyses can be selected, while the buttons of the ribbon *Analysis* task correspond to the selected analysis of the *Analyses* tree!

6 Optimization

This chapter gives a detailed description of the *Optimization* tree and of the required work-flow to perform an optimization task.

The software provides several possibilities to optimize a scenario: the most common application is the optimization of a trajectory, in a sense of attitude control and phase duration. Another aspect that can be improved via optimization is the performance of the vehicle (i.e transmitted data volume, payload, etc.); this can be realized by acting on the trajectory or on the design parameters of the vehicle (e.g. transmitting frequency, tank filling ratio).
These goals are formulated by defining related constraints (i.e. boundaries that limit the permitted range of design values) and cost functions (to provide a weight on different aspects that describe the "optimality" of a solution). The software provides an extensive set of building blocks for that purpose, enough to handle most of the common space scenarios.

ASTOS provides an interface to state of the art solvers. These are mostly coupled to an in-house transcription method (*CAMTOS*) which is able to handle global and gradient based optimizers with both multiple shooting and collocation methods. This chapter provides the required information to properly set up the software for solving an optimization problem whereas theoretical aspects are presented in Optimization Theory and Description of Methods book.

The **Optimization** feature must be installed and enabled (from the root node of the *Modelling* tree) to set up a scenario for this task; not only to run an optimization, but also to define optimizable parameters, cost functions and constraints. Moreover, a license for at least one solver (*WORHP* is default) is required. In case the feature or the license is missing in the installation, please contact service@astos.de.

6.1 Workflow

In an ideal case, the workflow to solve an optimization problem with *ASTOS* consists of the following steps:

1. Modelling of the problem, i.e. setup of

- Environment and vehicle (Chapter 4 and Model Reference)
- Constraints (Section 6.2)

- Cost functions (Section 6.3)

2. Generation of an initial guess of the optimal solution (Section 6.4)
3. Simulation of the trajectory using the initial guess from the previous step (Section 6.5)
4. Analysis of the simulation results (Section 8.2.1)
5. Optimization of the trajectory (Section 6.6)
6. Analysis of the optimization progress (Section 6.12)
7. Display of final results (Section 8.2.1)

For real problems, however, these steps are usually of iterative nature as shown in Fig. 6.1 which depicts a typical sequence of user activities and the related *ASTOS* tools which support the optimization process.

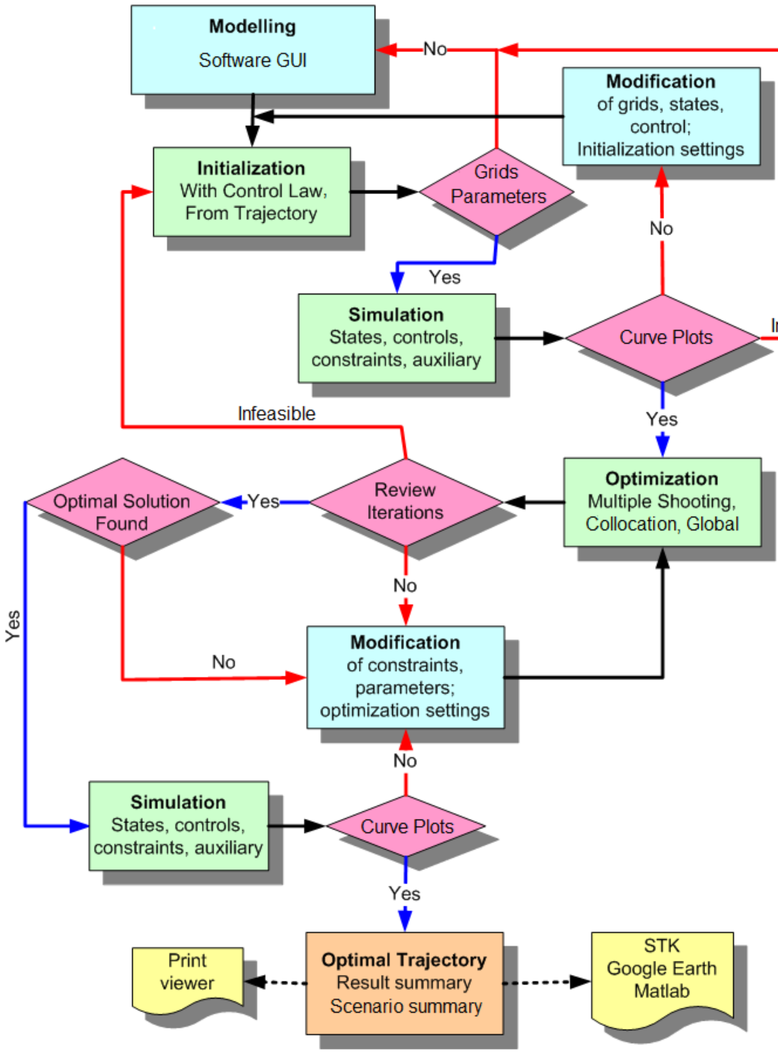

Fig. 6.1: Typical optimization workflow

- Black arrows identify the flow direction.
- Cyan rectangles identify modifications of settings or parameters

- Green rectangles identify *ASTOS* actions.
- Pink diamonds identify checkpoints where the user has to verify the correctness of the data.
- Blue arrows identify a positive answer.
- Red arrows identify negative answers.

The distinction between "No" and "Infeasible" is case-dependent and may require a certain degree of experience to interpret. An "Infeasible" solution could be non-integrable or the result could be physically meaningless in this case (e.g. the trajectory contains negative altitudes).

The optimization process is quite complex and comprises an interaction between the aerospace scenario and the mathematical solver. Details about the theoretical background is provided in the book Optimization Theory and Description of Methods. The key aspect to be considered is that a solver requires a good trajectory as starting point. This is the reason why a first loop of actions has be executed in order to generate a suitable initial guess.

It is important to note the **difference between initialization and simulation** actions: the first actions reads the model data and integrates the states to place the grids on the trajectory (only states and controls at grid nodes are saved); the latter action integrates the states at time steps as defined in the output settings and computes all the auxiliary functions required to properly visualize the trajectory.

Once the optimization loop starts, it is possible to review the process online via *Review Iterations* (Section 6.12). The rationale behind is that an "error" (i.e. an improper setting) in the scenario modelling might lead to a slow and useless optimization run. Having the possibility to visualize intermediate results supports the user in identifying errors and stopping the optimization run if necessary.

If any *ASTOS* action is run, first a scenario configuration verification and a model initialization is performed. This means that each model present in the scenario is checked for compatibility and loaded in the software.

6.1.1 Behaviour of the Modelling Settings

At the beginning of each action like initialization, simulation or optimization the *Modelling* is scanned and the model is initialized. Depending on the action, some values of the *Modelling* are ignored or not. This section presents, what a user has to do, after a specific value has been changed, or where a value has to be changed to take influence in a specific situation. Remember, that many values defined in the *Modelling Tree* are also editable in the *Optimization Tree* (see Chapter 6). Since *ASTOS 6* a semi-automatic `TOPS` structure editor is provided. It allows to add/remove or reorder phases. In this way major structural changes can be done without an initialization from control law or trajectory.

There is a clear subdivision into following scenarios:

- A *Modelling* definition reacts on all actions without changing the TOPS structure, i.e. no problem initialization is necessary, a value is used as it is given.
- A *Modelling* definition reacts on all actions with changing the TOPS structure, but can be solved with a re-initialization. A re-initialization adds, removes or modifies elements of the TOPS, but never the values and bounds of optimizable parameters. Hence an optimized result is not destroyed
- A *Modelling* definition reacts on all actions with changing the TOPS structure, i.e. a problem initialization or a manual adjustment using the TOPS structure editor is necessary.

> **Note:** A scenario initialization overwrites all optimized values of parameters and an old result is lost!

- A *Modelling* definition reacts only on a problem initialization. Those values must be modified in the Chapter 6 during an optimization, since the values present in *Modelling* are used only for a problem initialization.

Table 6.1 assigns various definitions corresponding to the scenarios above. There are some modifications, which may result in very complex model changes or which have no impact on the computations. To divide between both cases it is necessary to analyse the modifications from the optimization problem point of view. If optimizable parameters (states, controls or design parameters) are added to or removed from the scenario, a scenario initialization or manual structure modification cannot be avoided. On the other side the complete model definition can be redefined without new scenario initialization. For example:

- It is possible to split an *Auxiliary* vehicle component into two, to replace an *Aerodynamics* or *Environment* model completely, or to exchange the provided data tables of propulsion systems. See the documentation of the various models, if they define optimizable parameters.
- It is possible to replace e.g. an *Auxiliary* component by a *Payload* component. The mass multiplied by the *Payload scaling factor* should be equal the original mass. The bounds of the *Payload scaling factor* can be defined as needed. Replace the *Auxiliary* component by the *Payload* in the scenario definition. After a **Re-initialization** the *Payload scaling factor* has been added. A simulation results in exactly the same.

Table 6.1: List of Modelling definitions and their reaction on the model computations depending on the action

Modelling Definition	**Remarks**	**Without Probl. Init.**	**Needs Reinitializatio**	**Needs Problem Initialization**	**Only Problem Initialization/ Structure Editing**
Cost Function					
Scaling		X			
New type			X		
Add a cost function	If you add a cost function that requires	X			

Modelling Definition	Remarks	Without Probl. Init.	Needs Reinitializatio	Needs Problem Initialization	Only Problem Initialization/ Structure Editing
	parameters (e.g. Chebychev), *ASTOS* automatically reinitializes the phase that is active				
Constraint					
Add/Remove			X		
Subtype	Changes the computation!	X	(X)		
Limit value		X			
Limit type	Equality flag must be set manually in the *Constraints Info* without initialization			(X)	
Is Enforced flag	In Modelling			(X)	
Is Enforced flag	In Constraints Info	X			
Description			X		
Phase Configuration					
Phase times	Can be changed in GUI without problem initialization			(X)	
Model scheduling	May be very complex				(X)
Attitude Control Type	Very complex			X	(X)
Adding a phase	Verify manager window asks to confirm the phase	Only simulate			
Activating a phase	Verify manager window asks to confirm the phase	Only simulate			
Changing the initial state type or values				X	
Activate a propulsion system	Only that face is automatically reinitialized	X			

OPTIMIZATION

Modelling Definition	Remarks	Without Probl. Init.	Needs Reinitializatic	Needs Problem Initialization	Only Problem Initialization/ Structure Editing
Activate a maneuver at beginning of phase	Only that face is automatically reinitialized??	X			
Attitude control laws					
Profile	Not optimizable	X			
Req. Velocity param.	Not optimizable	X			
Target_Orbit param.	Not optimizable	X			
Dyn. Pressure Controller parameter	If not optimizable	X			
All others	Optimizable	X			
Model Definitions (e.g. Aerodynamics, Propulsions)					
Add/Remove	May be very complex, see this section				
Type	May be very complex				
Interpolation Data	If not connected to optimizable values, e.g. a profile of an optimizable control	X			
Values	If not connected to optimizable values	X			
Values	If connected to optimizable values, can be changed in GUI without problem initialization			X	X
Lower and upper bounds	Can be changed in GUI without problem initialization			X	X
Changing environment models	It is enough to simulate	X			
Mission Definition					
Add an auxiliary state				X	

Modelling Definition	Remarks	Without Probl. Init.	Needs Reinitializatio	Needs Problem Initialization	Only Problem Initialization/ Structure Editing
Change the mission type		X			

6.1.2 Reconstruction using Initialization From Trajectory

If a problem initialization is necessary, the following steps must be done to obtain an initialized problem with approximately the same trajectory, controls and parameters as the previously optimized result:

1. Simulate the reference scenario with the *Start→ Actions→ Simulate Multiple Shooting* ribbon.
2. Export the simulation data to a file from which the scenario initialization is reading the control and state history, i.e. Export *GESOP Structure* type and *Struct* subtype.
3. Insert the path of this file in the *Initialize from file* field in the *Initialization Settings* panel of *Optimization* tree.
4. Copy the optimizable grid to the initial grid in the *Initialization Settings* panel, using the button **Copy grid from working TOPS file**.

> **Tip:** The **Copy grid from working TOPS file** is available only in each phase, not in the default setting. All control grids are unified to a single grid, not considering this may change the result!

5. Read the final or duration independent variable (e.g. time) of each phase in the *Phase Overview* node of the *Optimization* tree.
6. Insert these values in each phase of *Dynamics Configuration*, *Phases & Common Settings* node of *Modelling* tree.
7. Repeat the same procedure reading the real parameters (*Real Parameters* node of the *Optimization* tree) and inserting them in the *Vehicles Parts & Properties* node of *Modelling* tree. E.g. *payload_scale_factor*.
8. *Start→ Actions→ Initialize* ribbon.

6.2 Constraints

A constraint is a condition of an optimization problem that the solution must satisfy, i.e. a boundary that limits the "degree of freedom" of a quantity related to a trajectory. *ASTOS* covers both discrete constraints (initial/final boundary conditions and parameter boundaries) and constraints which have to be fulfilled over time along a trajectory (i.e. path constraints).

The *Constraints* node in the *Optimization* tree contains a list of all constraints defined in a scenario. This chapter and its subchapters presents the basic actions required to add a new constraint and to modify its data. A complete list of constraints included in *ASTOS* and their background is provided in the Phase-End Conditions and Optimization Constraints section of the Model Reference.

All constraints which are listed in the *Constraints* node and are assigned to a phase are transfered to the scenario *TOPS* file during *Initialization*. Only constraints available in the scenario *TOPS* file are visible to the solver during optimization.

Note: A constraint is evaluated and taken into account only during an *Optimization* run and not during an *Initialization* run.

Constraint Evaluation

A constraint part of the optimal control problem must generally be always fulfilled to achieve an optimal solution. The constraint evaluation depends on the constraint kind:

- **Equality** constraints must be zero. The only accepted tolerance is the *Constraint tolerance* (typically in the order of *1.0e-6*), which is configured in the *Optimization Settings* and explained in Section 6.6.1. One example of equality constraints are so called point constraints. The constraint evaluation can also be enhanced with a *Reference* value. In this case, the constraint evaluation is reformulated such that the value of the quantity is forced to equal the reference value (instead of zero).
- **Inequality** constraints must be greater than or equal to zero. Thus, any positive constraint value fulfills the constraint. Slightly negative values are accepted as long as they are within the specified *Constraint tolerance* (please see Section 6.6.1). An inequality constraint gives more flexibility to the optimization problem than an equality constraint.

Add a constraint

There are two possibilities to add a constraint:
1. Select ribbon *Add*→ *Optimization*→ *Constraint*.
2. If a constraint of the desired subtype already exists, it is possible to clone it by right-clicking on the existing entry and selecting *Clone*. Then right-click the cloned constraint and press *Rename* to redefine the name of the new constraint.

When adding a new constraint, first a unique *Identifier* has to be specified. Then the desired *Type* and *Subtype* can be selected from drop-down list. A complete list and description of all individual constraints is provided in the chapter Phase-End Conditions and Optimization Constraints of the Model Reference.
After a constraint has been created, its *Type* and *Subtype* (e.g. *Altitude*) cannot be changed. To do so, a new constraint has to be created again with the proper subtype (and the old one

should be deleted). However, the specification of how a constraint is evaluated (at the initial or final time of a phase or as a path or parameter constraint) can be modified at any time.

Constraint outputs

Each path constraint creates an auxiliary output function only in the time range of the specified phases. This function is included as *Path Constraints* data in the *Simulation Data* list available from *Curve Plots* (Section 8.2.1) and *Condition Plots* (Section 8.2.1) in the *Results* tree and can be displayed in plots. A profile path constraint creates an additional output function for the limit value. Scalar constraints outputs (e.g. initial boundaries) can be used in reports (see Section 8.4) or are included the result summary (Section 8.5).

Required updates during optimization

All information of the settings defined in *Constraints* is saved in the `TOPS` file during the initialization (). It is not required to initialize the scenario after a new constraint has been added; the Verify Manager feature (Section 2.7) identifies the change also during a simulation or an optimization.
In case any settings of an existing constraint are modified in *Constraints*, only the information not already present in *Constraints Info* (Section 6.9) is automatically updated (e.g. the reference value). All the information present in *Constraints Info* (e.g. enforced or switching between reference and lower/upper limit) needs to be manually modified in *Constraints Info*. This manual adaptation is necessary only between two optimization runs (as the initialization would overwrite the results found in the previous optimization).

Tip: Rule of thumb: if a setting is present in *Constraints Info* **and** in *Constraints*, it should be changed in *Constraints Info* because the optimization action uses the information specified there.

6.2.1 Constraints Data

Description

During the optimization process in *ASTOS* the values of constraint functions and their scaled or unscaled violation are computed. For this three main kinds of constraint data management are supported. The first possibility is a single reference value, the second a finite range defined by a lower and an upper limit. Third is the profile data, which defines a limit by an interpolation table.

Four interfaces are supplied for declaration of all data formats:

- *Reference*, *Lower Limit* and *Upper Limit* value
  The constraint function has one parameter r which defines the target value of an equality constraint, or the limit of an inequality constraint. The corresponding constraint formulation is

$$f \begin{matrix} \leq \\ = \\ \geq \end{matrix} r \tag{6.1}$$

where *r* is given by input data and the constrained quantity is *f*.

- Reference *Range*
  The constraint function requires two parameters r_{min} and r_{max} which define the upper and lower limit of the constrained quantity, respectively. The corresponding constraint formulation is

$$r_{\min} \leq f \leq r_{\max} \tag{6.2}$$

  where *f* is the constrained quantity and both r_{min} and r_{max} are given by input data.

- *Profile*
  for some special constraints (e.g. *Alpha Machnumber*) the *Reference*, *Lower Limit* and *Upper Limit* is defined by a 2-dimensional boundary *y(x)*. The corresponding constraint formulation is

$$f \begin{matrix} \leq \\ = \\ \geq \end{matrix} y(x) \tag{6.3}$$

  where *f* is the constrained quantity and the boundary function *y(x)* is determined by interpolating tabular input data (x_i, y_i) provided by the user.

Tip: In contrast to the constraint type itself, the data type can be changed before any program call. In this way a single limit constraint can be changed to a range constraint. But if the equality is changed from reference to lower, upper or range and vice versa, then the equality flag has to be changed using the *Constraints Info* according to the limit definition if no initialization is performed.

The constraint reference/limit value can be changed without any need to initialize.

6.3 Cost Functions

The *Cost Function Terms* node includes the list of all the cost functions defined in a scenario. Together with some special weight factors related to smoothing, these items defines the objective function of the scenario. During an optimization task the solver tries to minimize the objective function while minimizing also the constraint violations. Consequently, the two goals for the solver are the objective function and the constraint violation. Only a correct scaling between these two goals ensures a feasible result with the lowest objective function.
Generally, three different cost functions classes exist:

- An **Initial cost function** is evaluated at the independent variable's initial value of a certain a phase. An initial cost function is a Mayer cost term.

- A **Lagrange cost functions** is evaluated at each major grid node except at the initial and at the final ones and these values are integrated over the interval of the independent variable for one phase.
- A **Terminal cost function** is evaluated at the independent variable's final value of a certain phase. A terminal cost function is a Mayer cost term and is also called final cost function.

This chapter presents the basic actions required to add a new cost function and to modify its data. A complete list of the available cost functions and their configuration and background is provided in Cost Functions of the Model Reference book.

Add a cost function

There are two possibilities to add a cost function:

1. Using the *Add→Optimization→Cost Function* action button in the GUI.
2. If a cost function of the desired type already exists, it is possible to clone it by right-clicking on the existing entry and selecting *Clone*. Then right-click the cloned entry and click *Rename* to redefine its name.

While adding a cost function, a unique name has to be inserted to identify it in the scenario. This *Identifier* can contain letters, numbers and underscores, but it cannot start with a number (`1st_max_payload` is not a valid name). In this case, an error is raised. Several *Types* and *Subtypes* are available to precisely specify the cost function (e.g. *Max Final Value* for *Altitude*). Once a cost function has been created, the *Type* and *Subtype* cannot be changed. Instead a new cost function needs to be created while the old one should be deleted, or alternatively, its *Scaling* value can be set to *0.0* instead.

Modifications between optimization runs

It is also possible to add or remove a cost function between two optimization runs. If the new cost function is different from the previous one, Verify Manager (Section 2.7) opens to inform about the identified modification (no user interaction is required). The feature automatically activates the currently defined cost functions in the proper phases and adds/removes additional parameters and constraints as required.

6.4 Initialization Settings

The *Initialization Settings* in the *Optimization* tree are used to customize all parameters needed to initialize the model and the the initial guess generator. In particular, this related to the grids creation, the trajectory integration with the provided control and the storage of information in the nodes.

The subchapters in this chapter explain how the *Initial Grids*, *Integration* and *Initial Guess* can be configured and why special attention is required in combination with the definition of conditions at the end of a phase.

Buttons

Initialize starts the initialization procedure. Alternatively the ribbon *Start→Initialize* action button can be used for that purpose.

Revert opens a dialog which provides the possibility to revert all settings to the either *Default values* for the initialization or to use *Saved backup values*. The latter correspond to the values saved in the previous session.

Update Model Data can be used to correctly update and display the current phase names in the *Initial Grids* panel. This action is extremely important to assign the grid setting to the correct phase.

> **Tip:** Press **Update Model Data** in case the phase list in *Initial Grids* contains "undefined" phase names (e.g. *Phase X* where X is an integer number) after a new phase has been added to the scenario. This links the grid settings to the new phase(s) defined in the *Modelling* tree.

6.4.1 Grid Types, Integration Settings and Initial Guess

The *Initial Grids* panel is used to define *Integration* and *Initial guess* settings. Furthermore, the *Major Grid*, *Control Refinement* and *Constraint Evaluation* for the optimization can be specified here. This covers both default settings which are valid for all phases and settings which are specific to a certain phase only.

Integration

The default or phase-specific *Integration* mode can be specified in the lower left part of the configuration panel.

> **Tip:** To activate a phase-specific setting, select a phase in the *Initial Grids* list and tick the checkbox relative to *Integration*. The name of the *Integration* panel then reflects the selected phase, e.g. *Integration (Lift_Off)*.

Four *Integration methods* are available:

- *Dormand-Prince-45* is a variable step size Runge-Kutta-Fehlberg integrator of fifth order which uses the Dormand and Prince formula.
- *Dormand-Prince-78* is similar to above but of the eighth order.
- *Runge-Kutta-4* is a fixed step size method of fourth order.
- *MEBDFDAE* (Modified Extended Backward Differentiation Formulae, Differential Algebraic Equations) is a variable step size integrator to solve stiff initial value problems based on [23].

Integration error is used for the internal step size control of the integrator. Demanding high accuracy results in long integration times. For most systems and applications, the default value of 10^{-8} is a good compromise between accuracy and computation time.

Note: Control laws which are computed depending on a dynamic system state often show sharp bends or even jumps at certain points of the trajectory. This may force the integrator to take extremely small steps.

Normalized step size specifies whether the provided *Minimum step size* and *Maximum step size* are normalized or de-normalized.

Minimum step size is a stopgap to avoid excessive computation times in case a control estimate that does not satisfy the smoothness assumptions of the integrator. This setting has no effect for fixed step size integrators. The default of 10^{-8} is suitable for most applications. The number refers to the normalized time of a phase, so the default setting forces the integrator to take at most 100 million steps.

Maximum step size is an optional setting to avoid the use of large step sizes during the integration of a phase. More details on that issue are provided in Section 6.4.3.

Ignore the minimum step size and just print a warning if the step size becomes smaller than the minimum is another optional setting. It can be ticked in case the fulfilment of the requested integration error has high priority and computation time is secondary.

Initial Guess

The default or phase-specific *Initial Guess* settings can be specified in the lower right part of the configuration panel. An initial guess can be created using either *Initialize from control law* or *Initialize from file*.

Initialize from control law (e.g. derived from optimality considerations) is used to integrate an initial trajectory using control laws defined for models in the scenario. In this case, the integration determines the state and the controls at the specified grid positions.

Note: Before starting an initialization, it is also necessary to have a basic `TOPS` file available, e.g. created via the *Optimization→ TOPS→ New* menu, or from `TOPS` file of another scenario.

Initialize from file imports the grid position and bounds from a file (`.txt` or `.struct`). This option uses states and controls and under certain conditions also real parameters and phase times defined in the file to generate a starting trajectory. The names in the file must be equal (case sensitive) with the names inside the model. If a state or control is missing in the file, the values are set to Zero. All bounds are taken from the *Modelling* data; if a value from file exceeds a bound, the bound is adjusted to the value from file.

- `txt`: the data are provided as tabulator separated list generated by the software or by an external tool. States and controls are read from the file; the phase times are taken only if a column *Phase* with the phase indices is present and the number of phases is the same. As example for a valid format, please use `simulation.txt` generated by the software.

- `struct`: it is the internal simulation data format of the software that has to be created in a previous run. States and controls are read from the file, as well as phase times and real parameters (if the ID are identical).

Major grid

The *Major Grid* describes the sequence of major grid nodes that are used for the optimization, i.e. either the collocation nodes or the multiple shooting nodes. The nodes at the beginning and at the end of a phase are generated automatically and can neither be specified nor deleted. These nodes are displayed in grey and are the built-in by default. In case of collocation, a high number of nodes is required to reproduce the states, please refer to Direct Collocation Method TROPIC in Optimization Theory and Description of Methods.

Tip: In case of multiple shooting a rule of thumb suggests to use no nodes for short phases (i.e. less than 50 seconds for a launcher), few nodes for long propulsive phases (i.e. one node every 100 seconds for a launcher) and few nodes for long coast phases (i.e. one node every 500 seconds for a launcher). Contact service@astos.de for other scenarios.

Control refinement

All major grid nodes are also control nodes by definition. When the major grid is too coarse for the controls, pure control nodes can be added by using the control refinement grid. This grid can be defined for each control independently by selecting the respective control branch below the *Control Refinement* node of a phase.

Note: This information is only used by multiple shooting methods. Collocation methods ignore any control refinement points.

Tip: A rule of thumb suggests to use about one node every 10 seconds in short phases where a complex profile is expected (i.e. after jettison of fairing) and less nodes for long phases with an expected smooth attitude profile (i.e. one node every 50 seconds during upper stage of a launcher). Contact service@astos.de for other scenarios.

Constraint evaluation

The *Constraint Evaluation* grid defines the points where path constraints should be evaluated.

Note: A constraint evaluation point at the initial or final phase time is ignored. Such constraints have to be represented as initial or final boundary constraint.

The constraint grid is ignored when used with *SOS* as optimizer which computes the constraint evaluation at each collocation grid point.

6.4.2 Grid Setup

To set up or modify a grid, first select the phase (or default setting) to be modified from the list, then activate the checkbox of the desired grid. Once a grid is selected, it is possible to insert or delete nodes using the panel on the right.

Tip: A selected grid is highlighted by a red frame, see control refinement in Fig. 6.2.

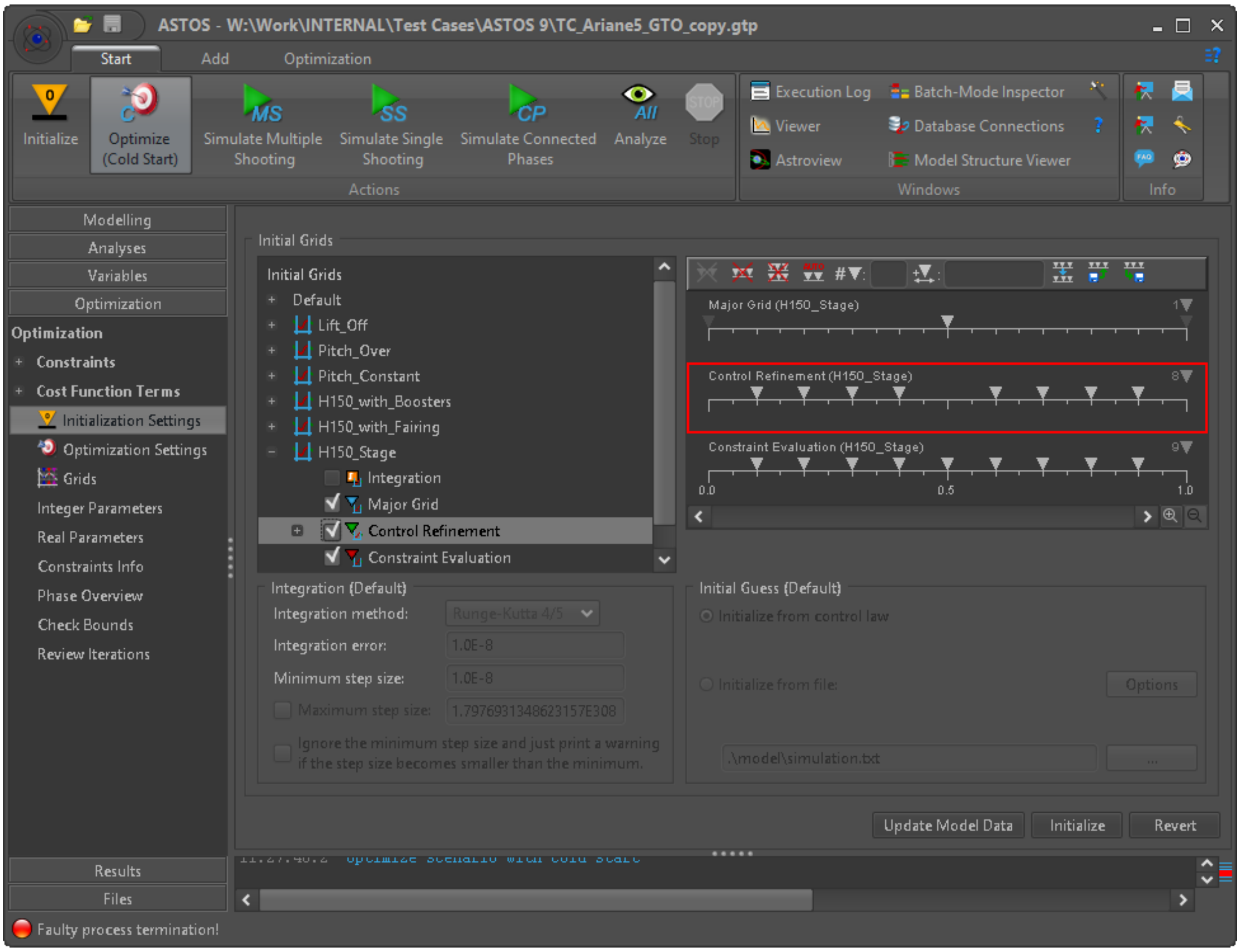

Fig. 6.2: Initialization settings in Optimization tree

There are several ways to add new nodes to a selected grid:

- Double-click at the desired node position on the displayed axis
- Insert the total number of equidistant nodes in the first input field of the panel toolbar
- Insert a normalized value (from 0.0 to 1.0) in the second input field of the panel toolbar
- Import the node positions from an `ASCII` file (see also below)
- Copy grids from the working `TOPS` file (not available for *Default* grids)

To modify a node position, select it (highlighted in red) and insert the new position as normalized value in the second input field of the panel toolbar. It is also possible to drag and drop a node to the desired position. A double-click on a selected node deletes the node. The actions available from the panel toolbar are:

- : Delete the selected node from the current grid.
- : Delete all nodes in the selected grid.
- : Delete all nodes in all phases and the default grid (a dialog appears for confirmation).
- : Activate/Deactivate the automatic grid generation mode (applicable only for collocation methods).
- Set a number of equidistant nodes in the current grid.
- Enter the position of a new node or modify the selected one in the current grid.
- : Copy grids from the working `TOPS` file (not available for *Default*).
- : Import grids from `ASCII` file.
- : Export grids from `ASCII` file.

Tip: Holding the mouse over a button displays a descriptive info message.

Automatic grid generation

When the automatic grid generation mode is selected (only for collocation method), a dialog appears after clicking the red "Automatic" label. Here, first an *Error tolerance* which defines the maximum error of the state approximation in comparison to the real state value can be specified. Furthermore the *Minimum distance* (normalized) between nodes can be specified. To deactivate the automatic grid generation, select the major grid and press the button again.

Tip: The automatic grid generation adds nodes to the existing grid. To use only automatically generated nodes, remove any major grid nodes before activating this feature.

6.4.3 Special Settings with Phase End Conditions

The scenario modelling allows the specification of phase end conditions (Section 4.5.1.6). This enables the integrator of the grid generation process (which is invoked by the action **Initialize**) to finish the integration somewhere between the lower bound and the nominal value of the final phase time as soon as the condition is fulfilled. This section describes how this behaviour can be influenced by user settings.

The point in time where the integrator stops (in case of a fulfilled phase end condition) and how long the integration process lasts, depends on the *Minimum step size*, the nominal phase duration, the number of grid points specified for the phase or alternately the *Maximum step size*. The minimum step size denotes the minimum possible time range to enclose the point in time, where the phase condition switches from false to true. If the accuracy of this condition is poor, the minimum step size needs to be reduced.

The grid generation integrates from one grid node to the next, where any node of the major and control grids is used. The integration itself is performed by the selected integration method. By default, this is a *Dormand Prince 5* integrator with a variable step size. In case the *Minimum step size* is reached, a simple Euler step is performed in order to step over a possible model discontinuity.

The definition of the grid has a specific influence on the integration especially in combination with phase conditions. The grid is specified by normalized time points. As a fulfilled phase end condition reduces the phase duration, also the normalization is changing and hence the position of the normalized time nodes. Due to this fact a second iteration with the new final time is required, which increases the integration time. In case the initialization is only used for simulation purposes, but not for optimization, no grid is required. In this case, the use of a *Maximum step size* leads to the same behaviour.

The phase end condition is checked at each integration step between the lower bound and the nominal value of the final time (in case no intermediate nodes are defined). Only if the phase condition is fulfilled at the final phase time, a more dense search for the switching point is initiated. In this case, it can occur, that the phase end condition gets true within this interval and false again before the final phase time is reached. A solution for this issue could be to specify intermediate nodes especially in regions, where the phase over condition is true. Alternatively, the *Maximum step size* can be used. Both methods force the integrator to perform smaller steps from the beginning.

Phase end conditions are ignored in case of normalized times. This raises the issue of overlapping phase times, i.e. due to adapted final phase times the bounds of the next phases could overlap which would result in an exception. This problem can be handled by setting the upper bound of the phase at the nominal final time. When the nominal final time is reset by a phase end condition, its upper bound is simultaneously reset to the new final time as well. This prevent the overlap between the upper bound and the lower bound of the next phase.

Note: Phase end conditions are ignored in case of normalized times.

6.5 Integration Settings

ASTOS provides three different simulation modes which differ in the underlying integration strategy (i.e. treatment of grid nodes) as depicted in Fig. 6.3:

- MS: Multiple Shooting
- SS: Single Shooting
- CP: Connected Phases

The desired simulation mode can be selecting by using the corresponding button in the ribbon *Start→Actions* group. In case it should be triggered automatically, the method can also be selected in the *Automatic simulation options* for the phases (see Section 4.5.1.4).

If the **Optimization** feature is disabled, only a single button is available. The action triggered by this button is equivalent to initial guess and connected phases with normalized time.

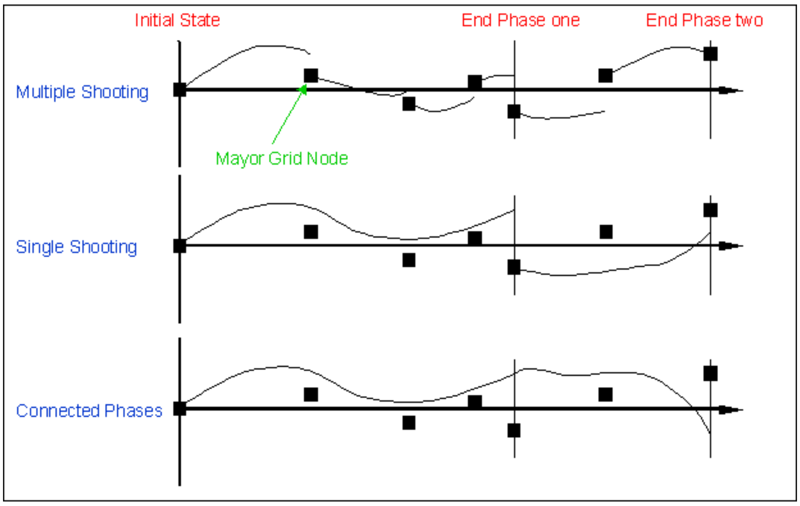

Fig. 6.3: The simulation modes (schematic)

Multiple Shooting

In this simulation mode, the integration stops at every major grid point and starts again with the value of the state vector at this node.
Given an initial guess or a not fully converged intermediate solution, there may be jumps at the major grid nodes. This simulation mode can therefore be used to detect discrepancies. This is also how multiple shooting optimizers "see" a trajectory, hence the name.

Single Shooting

In contrast to multiple shooting, the integration is carried out continuously within a single phase (in a "single shot") and the start value is taken from the optimization grid only at the beginning of a phase. This integration mode can be used to visualize phase connect discrepancies of a converged solution, e.g. caused by a coarse discretization in the case of collocation methods.

Note: This simulation mode coincides with the multiple shooting method in the first segment of each phase, as well as with the connected phases mode in the first phase.

Connected Phases

This mode only uses the initial value for the state of the first phase and then integrates over all phases.
At each phase boundary, phase connect conditions are evaluated and used as new start values. Only for states and parameters without connect conditions, the start values are taken from the available grid. With this mode, overall deviations of the trajectory can be computed.

6.6 Optimization Settings

Various parameters may be set to control the runtime behaviour of the optimizer and the underlying NLP-solver. Therefore, these input parameters are collectively referred to as **runtime parameters**. They control for instance, when an optimization stops either because stopping criteria are fulfilled or the maximum number of the permissible number of iterations is reached. These parameters also influence the amount of information presented to the user in the diagnostic output log.

In the following, runtime parameters are presented in more detail. They can be grouped into general, specific and expert user parameters. For all parameters, a default value is assumed if they are not explicitly specified. *General Options* coincide for all optimizers. This has the advantage that general runtime parameters only have to be set once, even if an optimization problem should be solved with several optimization codes. *Specific Options* belong to explicitly to an optimizer or its NLP-solver. **Expert user parameters** can only be addressed by optional solver input files (e.g. `SPECS` for *SNOPT* and *SOS* and `XML` for *WORHP*). This option should only be considered by expert users with a very good knowledge of the NLP methods.

Buttons

The buttons at the bottom of the *Optimization Settings* configuration panel can be used to directly start an optimization by pressing **Cold Start** or **Warm Start**. Alternatively, also the ribbon *Start→Optimize (Cold Start)* action button can be used. **Cold Start** is the default optimization action; **Warm Start** can be used only after the user has stopped a previous run and it will continue the optimization task as if it was never stopped. This is faster, but it is less efficient because the previous scaling factors are used.

Note: Warm start is not supported in CAMTOS. In this case use always Cold start.

The button **Revert** opens a dialog which provides the possibility to revert all settings to the either *Default values* for the optimizer or to use *Saved backup values*. The latter correspond to the values saved in the previous session.

6.6.1 General Options

The *General Options* are independent of the selected optimization method. All individual settings are described below.

Optimization method

ASTOS supports several *Optimization methods*:

- *TROPIC*: old transcription method supporting direct collocation, not to be used in standard situations.

- *PROMIS (SLLSQP)*: old transcription method supporting direct multiple shooting, to be used only for dense problems (i.e. density of Jacobian matrix above 30%).
- *PROMIS (SNOPT)*: old transcription method supporting direct multiple shooting, not to be used in standard situations.
- *SOS*: advanced transcription method supporting direct collocation, to be used only for problems with high number of optimizable parameters (i.e. above 5000).
- *CAMTOS*: default hybrid optimizer that supports direct multiple shooting and direct collocation, to be used in most situations.

Depending on the selection, different optimizer-specific settings are available which are further described in the following subchapters. Switching between a collocation and a multiple shooting method may require a verification of an appropriate initial grid; more details are provided in the *Major Grid* section of Section 6.4.1.

Max. iterations

Max. iterations specifies the maximum number of iterations performed by the optimizer - if convergence is not already attained before. This settings usually serves as a safe guard in case the problem is ill-posed or the initial guess is not within the convergence region of the optimization. If one is confident that the specified problem converges, then the default value for *Max. iterations* can be decreased (only in rare cases a problem requires more major iterations to converge). However, *SOS* should be used with a larger number of iterations (>500) as this method also considers an internal stop criterion.

Optimization tolerance

This value is passed to the NLP solver and specifies the convergence criteria for the **dual infeasibility**. This parameter is of primary importance: it controls how long the nonlinear programming method iterates until the stopping criteria (e.g. the Karush-Kuhn-Tucker (KKT) conditions in case of *SNOPT* or the Projected Gradient in case of *SOS*) are satisfied. Ideally, the KKT conditions should be exactly zero for an optimal solution. However, since problems are solved with a computer of finite precision, "zero" needs to be expressed in a numerical sense. Generally, one cannot aim at achieving a solution at an accuracy less than the accuracy of the computed numerical gradients. But even the latter accuracy is difficult to reach without considerable computational overhead. It is recommended to choose a tolerance value around 10^{-5}.

> **Tip:** *Optimization tolerance* values should not be less than 10^{-6} and typically not larger than 10^{-4}.

> **Tip:** The *Optimization tolerance* should be one to two order of magnitude larger than the *Relative integr. error* that is available in the *Specific options* for most optimizers (except *TROPIC* and *SOS*).

Constraint tolerance

This value is passed to all optimization methods (except *PROMIS*) and specifies the convergence criteria for the **primal infeasibility**. This corresponds to:

- The constraint violation (in case of *CAMTOS* and *TROPIC*);
- the equality and inequality constraint violation (in case of *SOS*).

It is recommended to choose a *Constraint tolerance* value between 10^{-5} and 10^{-6}. In case of a collocation method, the value depends also on the number of specified collocation nodes. A fine discretization grid generally permits a higher accuracy (e.g. for *SOS*, 10^{-8} could be achieved). *PROMIS* sets this value internally to 10^{-6}.

Iteration output frequency

This controls the frequency at which data is generated for the display in the *Execution log* (see also Section 2.3) and for the *Iteration review plots* (see also Monitoring Optimization Iterations in Optimization Theory and Description of Methods for a complete description of the optimizers' output). A certain value *i* means that data is written only for every *i*-th iteration (e.g. a value of 1 implies data output for each single iteration).

Note: In case the optimization is fast and each iteration takes few seconds, the *Iteration output frequency* for *Iteration review plots* should be increased to value larger than 1. Otherwise the GUI cannot manage to display the data properly.

Real workspace size/Integer workspace size

The remaining mandatory parameters, namely *Real workspace size* and *Integer workspace size*, control the available memory. Since the optimizers are not capable of allocating memory dynamically, it is necessary to provide a float and integer workspace in order to solve problems of varying (and principally unlimited) dimensions.

If the specified workspace sizes are too small for a particular problem, the optimizer provides an appropriate diagnostic message which informs about the minimum required size for each workspace (see also Error Codes in Optimization Theory and Description of Methods). If the specified workspace cannot be allocated by the system as it is too large, a `STORAGE_ERROR` occurs.

Typically, a *Real workspace size* (in 8-byte unit) of 300000 and an *Integer workspace size* (in 4-byte unit) of 200000 suffices for most problems, unless a very fine discretization is employed. *SOS*, for instance, requires much more workspace and values in the range of 1000000 are not unusual.

Note: *SOS* requires further memory during a mesh refinement. If the workspace is to small at this point, the optimization run stops without saving any result.

With *CAMTOS*, the provided values are uncritical: the optimizer automatically allocates workspaces that are large enough for the solver. In case user-defined values exceed internally calculated workspace sizes, the user specified numbers are used instead.

6.6.2 TROPIC Specific Options

Diagnostic output level

This parameter controls the format and content of the diagnostic output log. The desired level of output can be chosen from a drop-down list:

1. *Short Iteration Info* contains concise iteration info and the final SQP status
2. *System/Final/Collocation*: contains additionally optimal system status, overall final information and collocation constraint information
3. *Full Iter/Bnd Const*: contains additionally detailed iteration information and boundary constraint information
4. *All Constraints/Dynamic*: contains additionally path constraint, state and control information
5. *Complete Info*: contains additionally cost function, parameter constraint, integer and real parameter information

Note: The generated output is cumulative from top to bottom of the list, i.e. the output of previous level(s) is always included.

Note: *Complete Info* generates a huge amount of diagnostics data and should be only used if necessary.

Algorithm control

This setting defines the "goal" of the optimization and has three different options:

- *Minimize*: minimize the objective function subject to all constraints
- *Feasible Point*: search for a feasible point only while ignoring the user's objective function
- *Feasible Exit*: minimize the objective function and make sure that the final solution is a feasible point

Parameter and function scaling

Parameter scaling controls how optimization parameters are scaled by the software before they are passed to the nonlinear programming (NLP) code that solves the finite dimensional optimization problem:

- *None*: No parameter scaling is applied.
- *Linear*: All optimization parameters (discretized states, controls, etc.) are automatically scaled according to a linear transformation rule that is based on the parameters magnitudes at the start of the optimization.
- *Affine*: All optimization parameters are scaled automatically according to an affine transformation that is based in the lower and upper bounds of each parameter.

Function scaling controls the *TROPIC* actions to scale functions (such as internal collocation constraints or user-defined cost and constraint functions) before they are passed to the NLP code:

- *None*: No function scaling is applied.
- *Collocation Constraints*: All internal collocation constraints are automatically scaled such that they correspond to the scaled states.
- *All Functions*: All collocation constraints are scaled and all user-defined functions are automatically scaled with a damped scaling factor.

Note: These two settings related to the parameter and function scaling are very important and their impact on the optimization behaviour should not be underestimated. Without function or parameter scaling it often occurs that the optimization does not converge at all, unless the problem is numerically well scaled. For a better understanding of the implemented automatic scaling methods and related mathematical background information, refer to Parameter and Function Scaling in Optimization Theory and Description of Methods.

Difference Type

This option controls which formulation is used to compute the numerical gradients. Either *Forward* (Default) or *Central* differences can be used. Which of these formulations is more efficient is case-dependent.

Perturbation size

The *Perturbation size* is another parameter that controls the numerical gradient computation. It defines the relative size of the independent parameter perturbation used to compute the gradients of the cost and constraint functions. Depending on the selected *Difference type*,the default value is either the square root (for *Forward*) or cubic root (for *Central*) of the machine epsilon.

6.6.3 PROMIS Specific Options

The following list explains the different input parameters.

Diagnostic output level

This parameter controls the format and content of the diagnostic output log. The desired level of output can be chosen from a drop-down list:

1. *Short Iteration Info* contains concise iteration info and the final SQP status
2. *System/Final/Collocation*: contains additionally optimal system status, overall final information and collocation constraint information
3. *Full Iter/Bnd Const*: contains additionally detailed iteration information and boundary constraint information

4. *All Constraints/Dynamic*: contains additionally path constraint, state and control information
5. *Complete Info*: contains additionally cost function, parameter constraint, integer and real parameter information

Note: The generated output is cumulative from top to bottom of the list, i.e. the output of previous level(s) is always included.

Note: *Complete Info* generates a huge amount of diagnostics data and should be only used if necessary.

Algorithm control

This setting defines the "goal" of the optimization and has three different options:

- *Minimize*: minimize the objective function subject to all constraints
- *Feasible Point*: search for a feasible point only while ignoring the user's objective function
- *Feasible Exit*: minimize the objective function and make sure that the final solution is a feasible point

Note: When used with *SLLSQP* the algorithm control is always set to *Minimize* and cannot be changed.

Parameter and function scaling

Parameter scaling controls how optimization parameters are scaled by the software before they are passed to the nonlinear programming (NLP) code that solves the finite dimensional optimization problem:

- *None*: No parameter scaling is applied.
- *Linear*: All optimization parameters (discretized states, controls, etc.) are automatically scaled according to a linear transformation rule that is based on the parameters magnitudes at the start of the optimization.
- *Affine*: All optimization parameters are scaled automatically according to an affine transformation that is based in the lower and upper bounds of each parameter.

Function scaling controls the *TROPIC* actions to scale functions (such as internal continuity constraints or user-defined cost and constraint functions) before they are passed to the NLP code:

- *None*: No function scaling is applied.
- *Linear Weighted*: All internally generated continuity constraints and all phase connect constraints are automatically scaled such that they correspond to the internally scaled states. All other constraint violations and the cost are scaled relative to their initial value.

- *Pure Linear*: All internally generated constraints and all user-defined functions are automatically scaled with a damped scaling factor.

Note: These two settings related to the parameter and function scaling are very important and their impact on the optimization behaviour should not be underestimated. Without function or parameter scaling it often occurs that the optimization does not converge at all, unless the problem is numerically well scaled. For a better understanding of the implemented automatic scaling methods and related mathematical background information, refer to Parameter and Function Scaling in Optimization Theory and Description of Methods.

Relative integration error

This option controls the relative error tolerance for the integration method used to solve the ordinary differential equations. It has a major impact on the gradient precision, which must be larger than the required solution accuracy specified as *Optimization tolerance*. For the integration method used in *PROMIS*, the gradient precision can be expected to be one to two orders of magnitude less than the required relative integration accuracy. Consequently, it is required that the *Relative integration error* is at least two orders of magnitude smaller than the *Optimization tolerance*.

Absolute integration error

The *Absolute integration error* is another parameter that controls the numerical integration. It serves as a tie-breaker in case the trajectory encounters a zero crossing or if there is a "vanishing" trajectory (i.e. values converge to zero). A good guideline for the choice of the absolute integration error is the intended relative error or a value close to the machine precision, e.g. 10^{-15} (the default value of *PROMIS*).

6.6.4 SOS Specific Options

Number of mesh refinements

Num. mesh refinements defines the number of maximum mesh refinements (from 0 to 24) that are applied by *SOS*. During each mesh refinement, *SOS* adds additional major grid points to assess the accuracy of the approximation (doubling their number in maximum). Mesh refinements may be required to achieve the requested *ODE tolerance* (see below). Each mesh refinement adds one iteration to adapt the discretization and to repeat the optimization steps. The first refinement iteration uses the major grid nodes from the `TOPS` file.

Tip: In case a *SOS* error `MESH REFINEMENT ITERATION LIMIT` occurs or the optimization status is `Terminated after mesh refinement iteration limit reached`, the requested ODE tolerance is not achieved and the maximum number of mesh refinements is already reached. Either the number of mesh refinements has to be increased or the ODE tolerance may be relaxed.

Automatic scaling option

The natural variables and constraints as they appear in the optimal control problem can be scaled to improve the numerical conditioning and efficiency of the underlying nonlinear programming problem. Scale weights can be defined by the user and/or computed automatically. Mathematical details can be found in [1].

Algorithm control

This setting defines the "goal" of the optimization and has four different options:

- *Feas.-Minimize*: calculate a feasible solution first, then start to minimize the objective function subject to all constraints (Default).
- *Feas.-Min.-Stay Feas.*: search for a feasible point and start a minimization while staying feasible.
- *Minimize*: minimize the objective function subject to all constraints.
- *Feasible Point*: search only for a feasible point ignoring the user's objective function.

ODE tolerance

The *ODE tolerance* specifies the maximum local error to be achieved during the refinement iteration; i.e. the error between the polynomial approximation and the numeric integration. This tolerance is one of the optimality criteria.

Note: This tolerance has to be at least twice as large as the *Constraint tolerance*. A warning message appears if this condition is not satisfied.

Output options

Output options for *SOS* can be specified using the **Show sos.out Options** button. This opens a dialog where the desired *Initial Output*, *Iteration Output* and *Final Output* can be configured in detail. The `sos.out` is located in the scenario folder; this is typically deleted at the end of the optimization run unless the *Keep NLP solver output files* option is activated in the *Window→ Preferences*.

Tip: The *Diagnostic output level* settings are not available in *SOS*, but there is a link between the output saved in the file and the output printed in the *Execution Log*.

The settings present in *SOS.out Output Configuration* window are self explanatory, Automatic Row and Column Scaling considering the following correlation between SOCOUT flag and options label (Table 6.2).

Table 6.2: Flag vs. Option

SOCOUT Flag	Option Label
A	Print summary of the optimization problem
B	Print sparsity information of the Right Hand Side (e.g. dependency of states w.r.t. states)

SOCOUT Flag	Option Label
H	Print sparsity information of matrices (Jacobian, gradient, Hessian), results of an index set analysis and perturbation sizes
C	Print short transcription summary
E-F	Print index transcription from optimal control to NLP problem (Variable-Constraint)
D	Print summary of general SOS settings
G	Print scaling information
I	Print variable values
J	Print objective function value
K	Print constraint values and dynamic rates
M	Print collocation state approximation errors (ODE errors)
N	Print dynamic status
O	Print constraint adjoints
P	Print states & controls adjoints
S	Print grid refinement summary (CPU time, number of RHS calls & ODE errors)

The default settings is printing the complete output in the `sos.out` file; this could lead to a huge dimension especially in combination with a high number of parameters (e.g. more than one Gigabyte is not uncommon).

Tip: These settings are not affecting the output in the *Execution Log* window with the exceptions of when *Print variable values* is set to the first three options; there no output is printed.

Character workspace size

The *Character workspace size* is another workspace required for memory allocation of the b-splines transformations. Increase the value in case the optimization is terminated with `ERROR OCCURRED: Size of CSTAT: real workspace for the transformation to b-spline too small, please increase size of the character workspace size`.

Max. number of function evaluations

Max. number of function evaluations defines the maximum number of function evaluations during one optimization run. This value has to be increased in case of optimization status `Maximum number of function evaluations`.

Phase Specific Options - Integration method

It is possible to define integration methods individually for each phase (by selecting the respective checkbox in the list on the left of the configuration panel). *Default* settings are used for all unchecked phases.

At present, nine different direct transcription methods are available in *SOS*. All of them can be used to transcribe the optimal control problem into a nonlinear programming problem. Following the methods are briefly introduced. For a mathematical explanation, please refer to [1].

- *Euler*
  This is the simplest method and its use is restricted to very simple problems.
- *Runge-Kutta 2*
  This is a lower order Runge-Kutta discretization scheme. It is applicable to problems that only require low accuracy.
- *Runge-Kutta 3*
  Another lower order Runge-Kutta discretization scheme that can be used for low-accuracy requirements.
- *Runge-Kutta 4*
  A medium order Runge-Kutta discretization scheme. The NLP variables are the same as for *Herm-Simpson Com*.
- *Trapezoidal*
  This is the method recommended for simple applications since it is very efficient. Also for larger applications this method can proof itself valuable when it comes to finding a coarse solution during first mesh refinement.
- *Herm-Simpson Sep*
  Separated Hermite-Simpson is using third order polynomials which require states and control values at the midpoint of each interval. Thus, the number of NLP variables increases, which improves the accuracy and extends the computational complexity. The number of constraints is also increased since defects at the interval midpoints are evaluated.
- *Herm-Simpson Com*
  The Compressed Hermite-Simpson is using third order polynomials, but it requires only the values for the control variables at the midpoint of each interval. Thus, the number of NLP variables increases, but less than in *Herm-Simpson Sep*; as well for the accuracy.

6.6.5 CAMTOS Specific Options

Diagnostic output level

The **Select Diagnostic Output** button opens a dialog which allows a selection of specific parts of the output log to be displayed before and/or after an optimization using the respective checkboxes:

- *Show a summary of the optimization problem*: it is possible only before an optimization.
- *Show sparsity information*: it is possible only before an optimization.
- *Show a list of all violated constraints*: it is always displayed after an optimization; suggested also before it.
- *Show a list of all parameters*: it produces a lot of output, only for debugging purpose.
- *Show a list of all constrains*: it produces a lot of output, only for debugging purpose.

- *Write the Jacobian matrix to file*: it stores the matrix in a file `Jacobian` in the `.gtp` folder
- *Write the structure of the Hessian and gradient to file*: it stores them in two files `Hessian_Structure` and `Gradient_Structure` in the `.gtp` folder (Available only when using *SNOPT* NLP solver).
- *Show a list of the cost derivatives*
- *Show a list of the constraint derivatives*
- *Perform a derivative check*: the gradient of the functions are evaluated to identify discontinuities in the first derivative.
- *Two-line iteration output*: each iteration output is printed in two lines, this is the default format.
- *Detailed parameter information*: experimental status only.
- *Suppress all output*: no output is written to the *Execution Log*.

Parameter and function scaling

Parameter scaling controls how optimization parameters are scaled by the software before they are passed to the nonlinear programming (NLP) code that solves the finite dimensional optimization problem:

- *None*: No parameter scaling is applied.
- *Linear*: All optimization parameters (discretized states, controls, etc.) are automatically scaled according to a linear transformation rule that is based on the parameters magnitudes at the start of the optimization.
- *Affine*: All optimization parameters are scaled automatically according to an affine transformation that is based in the lower and upper bounds of each parameter.

Function scaling controls the *CAMTOS* actions to scale functions (such as internal constraints or user-defined cost and constraint functions) before they are passed to the NLP code:

- *None*: No function scaling is applied.
- *Linear Weighted*: All internally generated constraints and all phase connect constraints are automatically scaled such that they correspond to the internally scaled states. All other constraint violations and the cost are scaled relative to their initial value.
- *Pure Linear*: All internally generated constraints and all user-defined functions are automatically scaled with a damped scaling factor.

Note: These two settings related to the parameter and function scaling are very important and their impact on the optimization behaviour should not be underestimated. Without function or parameter scaling it often occurs that the optimization does not converge at all, unless the problem is numerically well scaled. For a better understanding of the implemented automatic scaling methods and related mathematical background information, refer to Parameter and Function Scaling in Optimization Theory and Description of Methods.

Perturbation size

The *Perturbation size* is a parameter that controls the numerical gradient computation. It defines the relative size of the independent parameter perturbation used to compute the gradients of the cost and constraint functions. It is suggested not to modify the default value.

NLP solver

The selection of the *NLP solver* and the respective options are defined in Section 6.6.5.1.

Phase specific options

It is possible to define different transcription methods individually for each phase (by selecting the respective checkbox in the list on the left of the configuration panel). *Default* settings are applied for each phase where no check-mark has been set.

As *CAMTOS* is a hybrid optimizer that supports direct multiple shooting and direct collocation methods, it has to be ensured that the grid specifications defined in the *Initialization Settings* are appropriate for the selected transcription methods. For phases that are optimized with a direct multiple shooting method, the same rules as for *PROMIS* can be applied. For phases that are optimized with a direct collocation method, the grid selection should be applied similarly as for *TROPIC* or *SOS*. Refer to Transcription Methods in Optimization Theory and Description of Methods for more details.

Integration method

The transcription method can be selected indirectly via the *Integr. method* setting. Currently, eight different transcription/integration methods are available for *CAMTOS*. Methods labeled with *Colloc* are direct collocation methods.

> **Tip:** The recommended standard methods are *Runge-Kutta 45* for direct multiple shooting phases and *Colloc Hermite* for direct collocation phases.

- *Paus (DMS)*
  A special second order integration method (for experimental use only)
- *Runge-Kutta 23 (DMS)*
  A lower order Runge-Kutta-Fehlberg integration method to be used in connection with problems that only require low accuracy
- *Runge-Kutta 45 (DMS)*
  A medium order Runge-Kutta-Fehlberg integration method that is applicable to most trajectory optimization problems. It is the default transcription method of *CAMTOS* and is comparable to the method implemented in *PROMIS*
- *Runge-Kutta 78 (DMS)*
  A higher order Runge-Kutta-Fehlberg integration method that can be used for applications with high accuracy demands, e.g. interplanetary trajectories (relative tolerance around 10^{-15}, absolute tolerance in the framework of the machine accuracy)
- *Colloc Hermite (DC)*
  A fourth order collocation method using Hermite-Simpson polynomials as a state time history approximation. The method is comparable with the method implemented in

TROPIC, but usually exhibits a larger convergence radius. It is considered as the standard collocation method in *CAMTOS*.

- *Colloc RK4 (DC)*
  A fourth order collocation method using one Runge-Kutta 4 step in each collocation interval (it can also be considered as a multiple shooting method without step-size control). The convergence behavior of the *Colloc Hermite (DC)* method is usually better.
- *Colloc Trapez (DC)*
  A second order collocation method which uses a trapezoidal approximation for the state time histories. This method can be used for optimizations with low accuracy demands.
- *Rodas 43 (DMS)*
  An integration method for stiff systems (for experimental use only)

Number of mesh refinements

Num. mesh refinements defines the number of automatic mesh refinements. *CAMTOS* supports a refinement of the collocation grid and - in case of multiple shooting - a refinement of the control grid. For each mesh refinement at most one node is added per grid interval (in the middle of the interval). In case of collocation the addition of one major grid node is associated with the addition of one node in the constraint evaluation grid. The collocation mesh refinement is based on a comparison with a precise integration, the control mesh refinement is based on the curvature of the discretized control. Each control grid is treated independently. A mesh refinement is initiated at the end of an optimization process, i.e. when the optimization would stop otherwise (without mesh refinement). After mesh refinement the optimization continues.
The refinement affects all the phases, so this setting can be defined only for the *Default* entry.

ODE tolerance

Specifies the tolerance between the collocation state approximation and a reference integration using a Runge-Kutta-Fehlberg 4/5 method. If the tolerance is exceeded, a new collocation node is added in the middle of the examined interval during a mesh refinement. The variable step-size of the Runge-Kutta-Fehlberg integration is controlled by the *Relative integr. error* value (see below).
The *ODE tolerance* affects all the phases, so this setting can be defined only for the *Default* entry.

Relative integration error

Relative integr. error controls the relative error tolerance used by the integration method to solve the ordinary differential equations. It has a major impact on the gradient precision, which must be higher than the required solution accuracy specified by *Optimization tolerance*. With the integration method used in *CAMTOS* the gradient precision can be expected to be one to two decimals less than the required relative integration accuracy. Consequently, the *Relative integr. error* has to be at least two orders of magnitude smaller than *Optimization tolerance*.

Absolute integration error

Absolute integr. error is another parameter which controls the numerical integration. It serves as a tie-breaker in case the trajectory encounters a zero crossing or if there is a "vanishing trajectory" (i.e. values converge to zero). A good guideline for the choice of the absolute integration error is the intended relative error or something near the machine precision like 10^{-15} (the default value of *CAMTOS*).

6.6.5.1 NLP Solver Settings

NLP solver

With *CAMTOS*, it is possible to choose from a list of different *NLP solvers*:

- *WORHP*: default gradient based solver, the best solution in most of the cases when no integer parameters are involved [24].
- *SNOPT*: gradient based solver, only in few cases better than WORHP [42].
- *MIDACO*: mixed gradient-global solver based on ant colony method [41].
- *PSO*: global solver based on particle swarm optimization.
- *CGA*: global solver based on constrained genetic algorithm method.
- *CSA*: global solver based on constrained simulated annealing.

Some solvers are dedicated to the solution of NLP problems (pure continuous) with the implementation of methods based on the gradient analysis: these should be used on problems where only real parameters and continuous functions are present; this is the case for most of the scenarios. In the cases where integer parameters are present, either a global method or a mixed one should be used: they are able to provide a global solution of smaller problems (lower number of optimizable parameter) and also the performance is typically reduced. More information about some of the solvers can be found in Nonlinear Problem Solver in Optimization Theory and Description of Methods, some of them have been added recently and the testing is still on-going; please contact service@astos.de in case of questions.

NLP solver options

The **Show NLP Solver Options** button opens a dialog which allows the selection of additional solver-specific options:

- In case of *WORHP*, the panel includes:
  - *Tolerance scaling factors for acceptable solutions*: they are factors multiplied to the tolerances defined in the *General Options* panel to specify the acceptable level; respectively *Feasibility tolerance* by *Constraint tolerance* and *Optimality tolerance* by *Optimization tolerance*.
  - *BFGS block size range*: they defines the minimum and maximum size of the quasi-Newton method to approximate the Hessian matrix.

- In case of *SNOPT*, the *Algorithm control* defines the goal of the optimization and can be selected from:
  - *Minimize*: it minimizes the objective function subject to all constraints.

- *Feasible Point*: it searches only for a feasible point while ignoring the user's objective function (cost).
- *Feasible Exit*: minimize the objective function and ensure that the final solution is a feasible point.

- In case of *MIDACO*, the panel includes:

  - *MIDACO stopping criteria*: these settings are self explainable and they address the most important parameter for a global solver; i.e. when to stop the optimization. The user has the possibility to act on the number of internal recovery actions without success (AUTOSTOP), on the optimization time (MAXTIME) and on the number of iterations without progress.
  - *Seed*: it is the initial seed for the pseudo-random number generator.
  - *Focus*: the higher the number the more it concentrates the solution search in the region near the best solution.

- In case of *CGA*, the settings allow the user to define the population dimension and evolution. Only a list is presented here, the explanation of the theoretical background can be found in Solving the Nonlinear Program with a Genetic Algorithm in Optimization Theory and Description of Methods.

  - *Population* defines the size and the stopping criteria as upper limit of generations with no improvements (stall).
  - *Fitness Scaling* convert the fitness scores based on constraints and cost function in a range that is suitable for the selection criteria; the conversion can be performed according to proportional methods or ranking ones. In case of *Top* the user has to specify the number of individuals that produce offspring.
  - *Selection* defines the operator that selects the individual for reproduction; available options are:

    - *Tournament*,
    - *Roulette*,
    - *SUS* Stochastic Universal Sampling,
    - *RSIS* Remainder Stochastic Independent Sampling.

    For all selectors the user has to specify the *Elite count*: the number of individuals copied into the next generation; whereas the *Tournament* requires also the number of random picked individual (i.e. size) and the probability to chose the best individual (i.e. probability).
  - *Cross-Over* defines how frequently (i.e. *probability*) and how two random generated parents are recombined to create an offspring. There are three operators for the real parameters (i.e. *Crossover function*) and four operators for the integer parameters (i.e. *Cross. fcn. integer*).
  - *Mutation* defines how frequently (i.e. *probability*) and how a parameter is mutated in one individual; the mutation can be *Uniform*, *Boundary* or *Gaussian*. The first operator is acting within the parameter bounds, the second place the value at a bound and the third

is mutating according to the user specified variance (i.e. *scale*) and shrinking factor as the generations go by (i.e. *shrink*).

- *Local Search* (SQP) can be used to improve the performance especially when equality constraints are involved. The user has to specify how frequent is activated (i.e. *Frequency* = 5 means every 5th generation), how many iterations are performed and the amount of population affected by the local search (i.e. *probability*).
- *Migration* defines how some individuals (loners) moves to better areas or more populated ones.

- The other solvers don't provide additional settings.

6.6.6 SPECS Files

SPECS files (`.spc`) allow to specify a lot of additional parameters to define the optimizer settings which are usually only required by expert users. SPECS files are plain text files, which consist of a sequence of directives separated by blank lines. These files can be defined for *SNOPT* and *SOS*. (the equivalent file for *WORHP* is `param.xml` [24]).

Note: The red message from WORHP `Error (Read_XML_File): Could not open file param.xml` should be treated as a simple warning that no file is provided and the default values are used instead.

SPECS Files for SNOPT

A SPECS file for *SNOPT* must be located in the scenario folder and must be named `snopt.spc`. Settings in the SPECS file overwrite settings specified via the GUI in *Optimization Settings*.

The example below shows the file content with all valid keywords and their default values. Keywords are grouped according to the function they serve for. A SPECS file does not need to contain all entries listed in the example, but only those, which are different from the default settings. Asterisks (*) mark comments which are also not mandatory in the file.

Some default values depend on ε, the relative precision of the machine being used. The values given here correspond to double-precision arithmetic on most current machines ($\varepsilon = 2.22\ 10^{-16}$). Similar values would apply to any machine having about 15 decimal digits of precision.

Examplary file content:

```
BEGIN * checklist of SPECS file parameters and their default values
* Printing

Major print level                 1          * 1-line major iteration log
Minor print level                 0          * no minor iteration log
Print file                        9          *
```

```
Summary file                     6          * typically the screen
Print frequency                  1          * minor iterations log on PRINT file
Summary frequency                1          * minor iterations log on SUMMARY file
Solution                         Yes        * on the PRINT file
* Suppress options listing

* Convergence Tolerances

Major feasibility tolerance      1.0e-6     * target nonlinear constraint violation
Major optimality tolerance       1.0e-6     * target complementarity gap
Minor feasibility tolerance      1.0e-6     * for satisfying the QP bounds
Minor optimality tolerance       1.0e-6     * target value for reduced gradients

* Derivative checking
Verify level                     0          * cheap check on gradients
Start objective check at col     1
Stop objective check at col      n1
Start constraint check at col    1
Stop constraint check at col     n1

* Scaling
Scale option                     1          * linear constraints and variables
Scale tolerance                  0.9        *
* Scale Print                               * default: scales are not printed

* Other Tolerances
Crash tolerance                  0.1        *
Linesearch tolerance             0.9        * smaller for more accurate search
LU factor tolerance              10.0       * limits length of multipliers in L
LU update tolerance              10.0       * the same during updates
LU singularity tolerance         2.0e-6     *
Pivot tolerance                  3.7e-11    * ε^(2/3)

* QP subproblems
Crash option                     0          * all slack initial basis
Elastic weight                   100.0      * used only during elastic mode
Iterations limit                 10000      * or 20m if that is more
Partial price                    1          * 10 for large LPs

* SQP method
Minimize                                    * (opposite of Maximize)
* Feasible point                            * (alternative to Max or Min)
* Feasible Exit                             * (get feasible before exiting)
Major iterations limit           1000       * or m if that is more
Minor iterations limit           500        * or 3m if that is more
Major step limit                 2.0        *
Superbasics limit                500        * or n1 + 1 if that is less
```

```
Derivative level               3          * assumes all gradients are known
Derivative linesearch                     *
Function precision             3.0e-13    * ε^0.8 (almost full accuracy)
Difference interval            5.5e-7     * (Function precision)^1/2
Central difference interval    6.7e-5     * (Function precision)^1/3
Violation limit                10.0       * unscaled constraint violation limit
Unbounded step length          1.0e+18    *
Unbounded objective            1.0e+15    *

* Hessian approximation
Hessian Full memory                       * default if n _ 75
Hessian Limited memory                    * default if n > 75
Hessian frequency              999999     * for full Hessian (never reset)
Hessian updates                20         * for limited memory Hessian
Hessian flush                  999999     * no flushing

* Frequencies
Check frequency                60         * test row residuals ||Ax-s||
Expand frequency               10000      * for anti-cycling procedure
Factorization frequency        50         * 100 for LPs
Save frequency                 100        * save basis map

* BASIS files
OLD BASIS file                 0          * input basis map
NEW BASIS file                 0          * output basis map
BACKUP BASIS file              0          * output basis map
INSERT file                    0          * input in industry format
PUNCH file                     0          * output INSERT data
LOAD file                      0          * input names and values
DUMP file                      0          * output LOAD data
SOLUTION file                  0          * different from printed solution

* Partitions of cw, iw, rw
Total character workspace      lencw      *
Total integer workspace        leniw      *
Total real workspace           lenrw      *
User character workspace       500        *
User integer workspace         500        *
User real workspace            500        *
END * of SPECS file checklist
```

SPECS Files for SOS

Similar to *SNOPT*, also *SOS* can be controlled by an optional SPECS file which overwrites *Optimization Settings* specified in the GUI. The SPECS file must be named `sos.spc` and must be located in the scenario folder. All possible values are listed in the *SOS* User's Guide [1].

One type of settings deserves closer attention: the statements which provide the measurement data information for *SOPE*.

Currently the GUI does not support the use of *SOPE*. Consequently, all required information needs to be included in `sos.spc` if *SOPE* should be used. It is necessary to state the paths and file names of the files that contain the measurement data for states and controls together. The handle is `gesop_sope_file`. Additionally, it is necessary that the file contains the weights for the calculation of the residuals. The handle is `gesop_sope_weights`. If no file for the weights is specified, the software normalizes each individual residual with its initial measurement value. `gesop_sope_states` is required to determine to which states of the problem description the provided measurement data belongs to. The same holds for `gesop_sope_controls` with respect to controls data. With all these handle, `sos.spc` content could look like:

```
#--Default settings for all phases
gesop_sope_file=model/src/gesop_data_20Hz_p1.txt
gesop_sope_weights_file=model/src/weights_p1.txt
gesop_sope_states=1,3,4,11,12,13
gesop_sope_controls=2,5
```

In a multi-phase setup the above settings would serve as default settings for all phases. To give phase-specific input, the phase number needs to be added to the handles preceded by an underscore. Below is an example for a setting for phase 2:

```
#--Phase specific settings
gesop_sope_file_2=model/src/gesop_data_20Hz_p2.txt
gesop_sope_weights_file_2=model/src/weights_p2.txt
gesop_sope_states_2=1,3,4,6,9,11,12,13
gesop_sope_controls_2=2
```

6.7 Grids

The *Grids* panel is used for editing all grid related aspects of the model. It is possible to edit the individual phase-specific state, control and path constraint evaluation grids. Furthermore, auxiliary functions and the Lagrange cost (if available) can be simultaneously displayed with abovementioned grids.
Several general and specific options provide the means for a fine-grained control over each of these grids.

Note: Directly after an initialization, the *Grids* settings coincide with the ones defined in the *Initialization Settings* (see Section 6.4.2). Modifications of the *Grids* settings do not affect the result of an initialization, but are only used for subsequent optimization steps. Any later initialization however overwrites all modifications of the *Grids* settings.

6.7.1 Grid Options

This chapter describes all settings and options in the *Grids* configuration panel.

Select an item for plotting

To select an item to be displayed or its grid to be edited, first select the appropriate *Phase* from the uppermost drop-down list in the configuration panel. The *Item* list below contains all the data present in the selected phase:

- *States*
- *Controls*
- *Path Constraints*
- *Lagrange Cost*
- *Auxiliary Functions*

Items of the first three categories contain grid nodes (see Section 6.4 for details about the initial grid generation).
To plot an item (as for *Curve Plots*, see Section 8.2.1) double-click on an item or drag and drop it to the table in the *Grid View Options* panel. Then press the **Show** button. If the *Viewer* (see Section 2.2) is already open, the plot is placed in the selected (red framed) section, otherwise the *Viewer* window automatically opens. **Show in New Tab** arranges the selected item in a new tab of the *Viewer* window. Furthermore, it is possible to change the y-axis of an item by clicking on its entry in the *Axis* column. and then choosing an axis from a drop-down list. While an item is selected in the list, it (or generally all items) can be deleted with the (All) button.

Any of first three item categories can serve as a standalone primary item. *Lagrange Cost* and *Auxiliary Functions* can only be used as secondary item together with a primary item. First one single primary item needs to selected and then optionally a secondary ones. In fact it is possible to add an arbitrary number of *Auxiliary Functions* to one single plot in the *Viewer*. Displaying these items in addition to a primary item, can be useful to compare two related quantities in the same graph.

Grid options

Once a state, control, path constraint or Lagrange cost) is selected, specific can be defined in *Options* section in the bottom right corner of the configuration panel:

- *Connected*
  The padlock indicates whether a **state or control** is connected to the previous phase. The status can directly be changed by clicking on the icon.

- The coloured lock () indicates a "soft" connection, i.e. the connection is managed by the optimization algorithm. As consequence, the connect condition may not necessarily be fulfilled during an optimization.
- The grey lock () indicates a "hard" connection, i.e. the connection is always fulfilled.
- The open lock () indicates no connection. This can lead to jumps between two phases.

The connection state can be also changed in the *Constraints Info* panel in the *Optimization* tree (Section 6.9).

Note: Hard-connecting the states between two phases ensures no discontinuity of a selected state between the phases. This can lead to a faster optimization and is useful for short phases (e.g. coast phases). However, hard-connecting also stiffens the optimization problem, and abusing hard-connecting states leads to a very stiff problem and likely infeasibility (since it is impossible for the optimizer to find a solution).

- *Continuous Control*
  This checkbox is only available if a **control** has been selected. The major grid point values and their bounds are divided in the middle (see e.g. Fig. 6.4, where the left half of the grid point is shown in red and the right half is shown in green). The reason for this is that a control may be discontinuous at a major control node and thus may have different values on the left and on the right side of a node. Obviously, the same holds also for the upper and lower bounds of such a grid point.
  It is currently not possible to modify the continuity on a grid point level. Instead the whole time history of the selected control item for a specified phase is affected by a change of the continuity. The continuity of a selected control item can be modified with the *Continuous Control* checkbox.
  If the flag is not checked, there is no immediate visual effect. This just flags the grid as discontinuous and the left and right node values of an item can be edited separately. This also allows the optimizer to treat these values separately as desired.
  If the flag is checked, a dialog appears to select how the continuity should be recreated. Choosing *Left Value* maintains the left values and bounds of all grid points and the right ones are adjusted accordingly. Choosing *Right Value* has the opposite effect. Choosing *Mean Value* modifies all grid points according to mean of left and right values respectively.
- *Approximation*
  In case of a **control**, the approximation method between grid points can be selected:

  - *Piecewise Constant*: The control function is approximated with constant value. Disconnected segments starting at each control node.
  - *Piecewise Linear*: The grid points are connected by straight line segments (default setting).
  - *Cubic Spline*: Approximates the control with cubic splines between grid nodes. Spline approximation is only useful for very special cases. *CAMTOS* does not support spline approximation.

- *Enforced*

The *Enforced* checkbox is available in case of a **path constraint**. The selection defines if the currently selected path constraint should be considered in the optimization or not. In the latter case however, its value is still computed.

Note: These phase-specific settings are used for subsequent optimizations and the respective general *Is Enforced* settings of the *Constraints* node are ignored for this constraint in the current phase. Any later initialization overwrites these phase-specific settings again!

- **Copy from State and Control Grids**
  This button (available in case of a **path constraint**) creates automatically a constraint grid which contains all grid points of the major grid and of all control grids.

Note: The existing constraint grid gets discarded.

6.7.2 Editing Grids

The initial setup of grids is already explained in Section 6.4.2. This sections describes how state, control and path constraints grids can be further edited (for optimizations after an initialization) when displayed from the *Grids* panel in the *Viewer*.

State grids

When **state grids** are displayed, the *Viewer* shows the values and the upper and lower bounds of the grid nodes for the selected quantity. The state nodes are depicted as yellow squares and the corresponding upper and lower bounds - as yellow triangles respectively. The color of the node at the beginning of a phase is blue and for the node at the end of a phase it is magenta. In addition, the function associated with the chosen state is shown as a grey line connecting the grid points if a simulation result is currently available.
On the plot abscissa at the position of the state grid nodes small triangular markers are also present. These markers identify the position of all grid points as a combination of all currently existing grids. If such a marker does not coincide with a visible grid point, this indicates that a grid point at this position is present in some other grid, e.g. a control refinement point or a constraint evaluation point. This feature is meant to give an overview of the current grid layout in order to avoid placing too many grid points at distinct locations which is not recommended from a numerical point of view.

Control grids

Editing **control grids** is analogous to editing state grids. The default display in this case also shows the nodes and the upper and lower bounds of the control grid points for the selected control quantity. The control functions do not depend on any simulation results and thus, in contrast to the state grid, the control functions depends on the *Approximation* setting as explained in Section 6.7.1 and can be piecewise linear or a cubic spline.
Recall that the control grid structure is more complex than the state grid. In each phase there is only one common state grid for all states which also defines the major control grid for all the

controls. However additional control refinement points may be specified separately for each control item where necessary as explained in Section 6.4.

Note: These refinement grids for each control are meaningful only for multiple shooting optimizers.

The modification of nodes for *Controls* is performed as for the *States*, however it is not possible to delete or modify the abscissa of major grid nodes (red-green nodes).

Constraint evaluation grids

A constraint evaluation grid is shared by all **path constraints** defined in a phase. All path constraints are evaluated point-wise at this discretization grid, but an evaluation point at the final/initial time of a phase is not considered. A constraint at these nodes must be defined as final/initial boundary constraint. This approach has proven to be numerically better behaved than the alternative integral approach, where the violations are integrated over time and a final boundary constraint is employed to assure constraint satisfaction.

Note: Path constraints are not evaluated at initial/final phase times.

Based on the path constraint formulation, a constraint is violated when its internally scaled value is less than zero. To improve the readability, constraints are however displayed unscaled in the plots (with red background in a forbidden area). As pointed in the note above, the optimizer "sees" a violation only in the evaluation point of the path constraint grid.

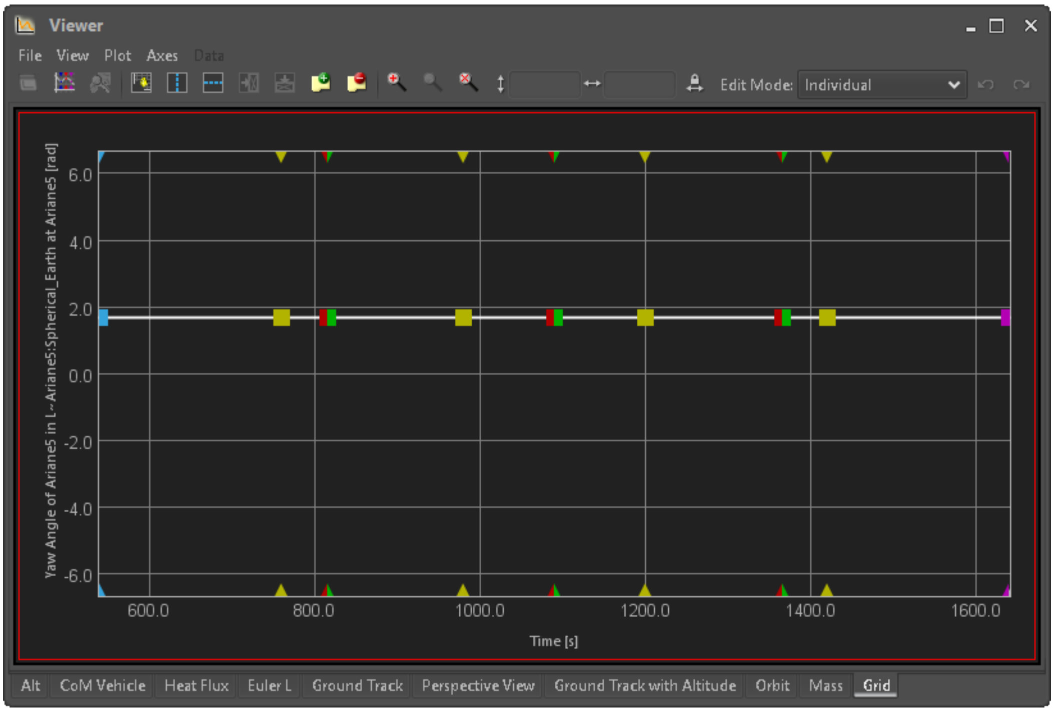

Fig. 6.4: Grid display in the Viewer

Move a node

The modification of a grid node value one of its bounds in the *Viewer* (Fig. 6.4) can be realized in two different ways:

- Select the appropriate node/bound symbol in the plot area with the mouse, hold the left mouse key and move the symbol up and down. When placed at the right location, release the mouse button. By default, only the ordinate value can be modified. To modify also the abscissa value, first the button in the *Viewer* toolbar must be pressed. For orientation, the numerical values of the current position are shown in the two fields next to the and icons in the toolbar.
- Select the appropriate symbol (node or bound) with a single mouse click. The symbol gets highlighted and the numerical values of its current position are displayed in the toolbar as described above. These values can be manually overwritten in the input fields to specify exactly a desired value. The *Enter* key confirms the modification.

Note: Nodes cannot be moved outside of its bounds in order to keep the grid data consistent.

Insert/delete a node

A new node can be inserted to a grid with a double-click at the desired position. Node values can then be adjusted as described above. Similarly, an existing node can also be removed from the grid with a double-click.

Tip: There is only one common grid for all states of a phase, so it recommended to adjust the node and bounds of all other state items as well.

Note: The first and last grid point cannot be deleted as they are mandatory for the optimizers. However, it is possible to modify their values and bounds.

Modify node relations

There are several selection-specific options to modify the editing behavior in case multiple nodes (or bounds) are selected.

Tip: To select multiple nodes, drag a rectangle around all desired nodes while the left mouse button is pressed. All selected nodes get highlighted with a red frame. The *Ctrl* and *Shift* keys can be used together with a mouse-click to add/remove certain selections similar as in the *Microsoft* file manager.

The *Edit Mode* can be selected from the *Viewer* toolbar:

- *Individual*: Each node is moved separately (default).
- *All Constant*: All selected grid points of a certain type (nodes or bounds) are set to a common constant value. First select the nodes (or bounds), then activate the *All Constant* mode and move one node to the desired position.
- *All Linear*: The selected nodes (or bounds) are placed along a straight line connecting the first and last selected node in the grid.

- *Bnds. Rel. to Pnt.*: All selected upper or lower bounds are placed at the same relative distance to their node values. This allows an arrangement of the bounds in a tube-like fashion around the current trajectory.
- *Bnds. Fixed to Pnt.*: The selected node is fixed by setting its upper and lower bound equal to its value. In this edit mode, bounds are automatically set accordingly when clicking a node symbol.

Undo/Redo

The ( ) icon can be used to undo (or redo) every action performed in the current *Viewer*.

Note: The action history is deleted every time an external action is performed, e.g. a new state is plotted or another tab was selected. In these cases the *Optimization→TOPS→Open Backup TOPS* action button can be used to revert the grids to the last saved TOPS version.

6.8 Integer and Real Parameters

The *Integer Parameter* and *Real Parameter* panels can be used to edit integer and real parameters of the model and their bounds. These parameters are either automatically created by certain models (e.g. propulsion models create parameters for ignition or shut off times) or they are created if the user specifies a parameter in a model as *Optimizable*. They are not displayed in the respective panels before an initialization.

Note: The optimization of integer parameters requires a global or mixed solver (e.g. *CGA* or *MIDACO*, see Optimization Theory and Description of Methods).

The configuration panel which is identical for real and integer parameters is described below.

Parameter configuration

The *Phase* drop-down list determines the phase whose parameters should be to displayed or edited. The table in the *Item Selection* panel then displays a list with the *Name*, *Value* and *Lower/Upper Bound* of all parameters in this phase.

Tip: Values and bounds may not be displayed with all digits as they are rounded for readability reasons. The exact value is displayed however, when double-clicking on an entry.

The column entries are used to indicate if or how a parameter is connected to value of the previous phase:

- The coloured lock () indicates a "soft" connection, i.e. the connection is managed by the optimization algorithm. As consequence, the connect condition may not necessarily be fulfilled during an optimization.

- The grey lock ( ) indicates a "hard" connection, i.e. the connection is always fulfilled.
- The open lock ( ) indicates no connection. This can lead to jumps between two phases.

The connection mode can be also be changed in the *Constraints Info* panel of the *Optimization* tree (Section 6.9).

Note: Hard-connecting the states between two phases ensures no discontinuity of a selected state between the phases. This can lead to a faster optimization and is useful for short phases (e.g. coast phases). However, hard-connecting also stiffens the optimization problem, and abusing hard-connecting states leads to a very stiff problem and likely infeasibility (since it is impossible for the optimizer to find a solution).

The column is used to indicate whether the respective parameter should be optimizable in the selected phase or not. If checked, the parameter is treated as optimizable and the lower and upper bound columns can be adapted. If unchecked, the parameter is considered to be static and the cells for the bounds are greyed out and ignored.

Note: When the flag is unchecked again, the defined bound values are lost as they are automatically set to the parameter value.

Tip: Modifaction of cell value need to be confirmed with the *Enter* key. The GUI checks whether the new value is legal and in case it is not, an error message pops up.

Protected parameter(s) modifiable defines the protection status of normalized time parameters. It is only available for *Real Parameters*. If unchecked, related parameters are greyed out and protected from modifications. Otherwise, the values can be manually change, but modifications are not checked for consistency.

Note: It is highly recommended to modify normalized phase time parameters in the *Phase Overview* panel of the *Optimization* tree (Section 6.10) and not in the *Real Parameter* panel.

Notes on phase connections

A phase connect is used to set the value of a parameter in a phase equal to the value of the same parameter in the previous phase. There are some special situations:

- The initial value of an attitude control angle set as *Linear Law* (e.g. *initial_PITCH*) is linked to the constant or final value of the same control angle of the previous phase (e.g. *PITCH* or *final_PITCH*).
- In case of normalized time the *Phase.Initial* is linked to the *Phase.Final* of the previous phase (see also Section 4.5.1.2).
- For each actuator, the *Virtual_Ignition_Time* parameter is linked to the previous phase considering the duration of coasting phases.

> **Tip:** When a connected real parameter is manually modified by the user, a dialog appears which proposes an automatic update of the connected real parameter (with the same name) in the other phases: **No** should be here selected in case of *Phase.Initial* parameters.

6.9 Constraints Info

The *Constraints Info* panel is used to edit the constraint settings in the scenario `TOPS` file after an *Initialization* task has been already performed at least once and prior to running an *Optimization* task.

The following settings are present in the panel:

Phase

The *Phase* drop-down list allows the selection of the phase whose constraints need to be modified.

Constraints

The *Constraints* list allows to select the type of constraint which need to be modified. The following constraint types are available:

1. *Initial Boundary Constraints*
2. *Final Boundary Constraints*
3. *Nonlinear Parameter Constraints*
4. *Path Constraints*
5. *Connected States To Previous Phase*
6. *Connected Controls To Previous Phase*
7. *Connected Integer Parameter To Previous Phase*
8. *Connected Real Parameter To Previous Phase*

The first four types are available in all phases. The last four constraints are not available in the first phase.

Item Selection

The *Item Selection* table lists all constraints for the *Phase* and *Constraints* type already specified.

The following icons are used in the table:

 *Is equal?* (for all constraint types)

 *Enforce* (for constraint types 1-4)

Connection to previous phase (for constraint types 5-8)

Soft connected

Hard connected

Not connected

The table consists of three columns:

- The *Name* column contains the names of all constraints in the chosen *Phase* and *Constraints* type.
- The *Is equal?* (icon) column allows to set if a constraint shall be treated as equality (selected) or inequality constraint (not selected).
- The third column header is an icon and depends on the selected *Constraints* type. There are two possibilities:
  - The *Enforce* column (for *Constraints* types 1-4 above) allows the user to select if a constraint shall be enforced (selected) or not enforced (not selected). In the latter case, it is ignored by the optimizer and only its value is computed. This allows a large number of constraints (even conflicting ones) to be considered in the model with different sets of constraints enforced in different steps of the optimization process.

Tip: The auxiliary functions associated to a path constraint are computed during the simulation process even if the constraint is not enforced. This functionality can be used to obtain a "quick-look" of complex path constraints (as e.g. *Station visibility* or *Splash down*).

  - The *Connection to previous phase* column (for constraint types 5-8 above) indicates how a quantity is connected to the value of the previous phase. The status can be changed by clicking on the icon in the respective cell in the row of the desired constraint:
    - The *Soft connected* icon indicates a "soft" connection, i.e. the connection is managed by the optimization algorithm. As consequence, the connect condition may not necessarily be fulfilled during an optimization.
    - The *Hard connected* icon indicates a "hard" connection, i.e. the connection is always fulfilled.
    - The *Not connected* icon indicates no connection. This can lead to discontinuities in the values on the boundary between two phases.

Alternatively, the connection status can also be defined in the *Integer*/*Real Parameter* panels (Section 6.8) and in the selection specific *Options* of the *Grids* viewer (Section 6.7.1)

Note: In order to enforce an already available scenario constraint between two *Optimization* tasks, the *enforced* property of a constraint has to be changed in the scenario TOPS file. This is done from the *ASTOS* GUI by *enforcing* the corresponding constraint in the *Constraint Info* panel, and not by *enabling* the *Is enforced* flag in the *Constraints* node.

6.10 Phase Overview

The *Phase Overview* provides an overview of the schedule of all phases defined in a scenario. The initial and final times of all phases as well as their corresponding bounds are presented on a timeline. The time values of the phases can be modified and adapted by the user.

Note: Modifications in the *Phase Overview* are applied to the TOPS file, i.e. they affect the simulation and optimization. Values present in the *Phases* configuration of the *Modelling* tree remain unaffected by such modifications and are only used to initialize a scenario.

The configuration panel of the *Phase Overview* is divided into three section: a general **toolbar**, the *Time Values* section to display all numerical values for the selected phase (e.g. Phase 5) and the graphical representation section of the phases timeline.

Toolbar

The toolbar contains the following functions:

- /: Undo/Redo changes applied to a phase time.
- : Revert to the default zoom level of the graphical representation.

Tip: To zoom into a certain part of the phases timeline (e.g. a very short phase), press and hold the right mouse button and move the mouse cursor to the side over the time interval of interest. The zoomed area can be adjusted with the lower scroll bar below the timeline graphical representation. At the right end of the scroll bar there a *zoom in* and a *zoom out* button to adjust the zoom level of the timeline graphical representation.

- An input field to specify a value for an item which has been selected in the timeline graphical representation. The field is preceded by an icon that indicates the selected item type (see below).

Time values and phase vs. time diagram

The *Time Values* panel displays all numerical values corresponding to a selected phase. A phase is selected by right-clicking on the phase number on the left side (*Phase* axis) of the timeline graphical representation. This causes that the selected number is highlighted in red. The following items are displayed in this panel:

- , , : Lower bound, nominal and upper bound values of a phase initial time
- , , : Lower bound, nominal and upper bound values of a phase final time
- , , : Lower bound, nominal and upper bound values of a phase duration. The phase duration values are only displayed if the scenario uses normalized phase time (see *General Settings* in *Modelling* tree Section 4.1).
- horizontal solid line: represents the duration of a phase
- horizontal dotted line: links the lower and the upper bound of the final time of a phase
- vertical dotted grey line: auxiliary line to quickly identify final values of phases

To modify the value of an item, two possibilities exist, while an automatic check of the limitations on these values in each phase is done:

- An item can be selected in the plot section with a left-click and dragged horizontally to the desired position. If the user tries to locate e.g. the final time outside the final time bounds then a yellow warning symbol appears and a brief explanation is displayed.
- An item can be right-clicked on in the *Time Values* panel and its numerical value can be then modified in the *Toolbar*. If the new value violates any limitation imposed by the phase, then a warning window appears and the value entered by the user is modified accordingly.

Tip: In case normalized time is used for a scenario, in order to change the bounds on the initial time of a phase, the bounds on the final times of the previous phase have to be modified. The reason for this is that the initial time of a phase is connected to the final time of the previous phase.

Phase Times Inconsistencies

A warning symbol is displayed when two inconsistent time values cause a duration conflict or a phase connect conflict. A duration time conflict is present when the phase final time is not consistent with the phase duration time (applicable only in case of normalized phase time). A phase connect conflict is present when the phase final time is not equal to the phase initial time of next phase.
A click on the yellow warning symbol in such a case opens a dialog window to confirm if the software should automatically solve the problem.

Note: Such inconsistency could be present, if a user modifies normalized time values in the *Real Parameter* panel (this is why they are protected by default): It is thus strongly recommended to modify time values only via the *Phase Overview*.

6.11 Check Bounds

The *Check Bounds* panel can be used to determine if a state, control, real parameter or a phase time is at its lower or upper bound after an optimization.

The *Settings* panel specifies items and options for a bounds check:

- *States / Controls / Real parameters / Phase time*:
  Determines for which quantities the bounds shall be checked.
- *Ignore fixed values*: Ignores above quantities where no or "zero" bounds are specified.
- *Bounds Slack Range*: By default, the procedure tests, if the value of a parameter is equal to one of its bounds. If this option is activated and the value is greater than 0% then it is tested, if the value is near to the bound, where the slack range is defined by: (Upper Bound - Lower Bound) * Slack Range / 100
- *Connected flags*: Determines if connections between items should be checked (e.g. phase connections)
- *Connection Defect*: This check recognizes connection defects between connected items of the same name. Again a slack range can be defined.

  > **Note:** Connection defects can only be detected correctly between items of the same class (e.g. state) and name and if the connection rule is a simple equality.

- *Opt. grid time bounds*

The **Do Check** button starts the check procedure throughout all phases and the result is displayed in the *Output* log. The items that run at either their lower, upper or both bounds are displayed for all phases.

> **Tip:** It is highly recommended to press **Do Check** at the end of each optimization run: an `Optimal solution found` statement with a parameter at its bound is not a real optimum.

The naming convention in the *Output* log correspond to the ones described in Section 8.1. Additionally each entity is preceded by its type (e.g. `Rpar` for real parameters) and a number in brackets which denotes the item number in the respective overview for a specific phase, e.g.

- `Rpar(2)` denotes the second parameter in the *Real Parameters Item Selection* table
- `Control(3)` denotes the third control in the *Grids Item* table

6.12 Review Iterations

The objective of *Review Iterations* is to monitor graphically a variety of iteration-related information generated by the optimization code. Quantities such as convergence criteria, cost functions and constraints, but also the iteration history of the states and controls can be displayed.

This chapter describes the *Review Iterations* panel, in particular how to select the content to be shown in the graphs. The methods how to modify the layout of plots are fully described in Section 2.2.

Note: *Review Iteration* cannot be used with the *CGA*, since this solver does not provide the required output files.

The *Review Iterations* panel can either be accessed by selecting the respective node in the *Optimization* tree or by selecting the button in the *Viewer* window. The following *Review Iterations Options* are available:

- *Scalar History Graph*
- *Grid History Graph*

6.12.1 Scalar History Graph

Scalar iteration data, as the name indicates, contains only scalar items concerning the optimization iterations. These are, for example, convergence criteria, cost functions but also real parameters, boundary constraints and the norm of collocation constraints of each state. The purpose of this graph type is to display the iteration history of a scala items in form of a line plot which connects the iteration values by piecewise linear line segments. This graph type requires the scalar data source `\gismo\scal_iter.dat` (usually created automatically).

The following tables lists all items, both general and phase-specific ones.

Table 6.3: General items supplied in the scalar iteration data file

Item ID	Remark
cost	Cost function value
const_norm	Maximum constraint violation
step_size	Line search step size
merit_func	Merit function value
merit_deriv	Merit function derivative
primal_infeas	Maximum primal infeasibility
dual_infeas	Maximum dual infeasibility
penalty_norm	Norm of penalty parameters
cond_Hz	Reduced hessian condition
sdir_norm	Norm of search direction
dof	Number of degree of freedom (only in *CAMTOS*)
min_iter	Number of minor iterations (only in *CAMTOS*)

Item ID	Remark
nswap	Number of column swaps in the BS factorization (only in *CAMTOS*)

Table 6.4: Phase specific items supplied in the scalar iteration data file

Item ID	Remarks
Initial Boundary Constraints	Initial boundary constraint violation
Real/Integer Parameter	Real/Integer parameter value
Time_f	Final phase time (only if optimizable and not normalized)
L2 Collocation Constraints Norm	L2 norm of all collocation constraints of states (only *TROPIC*)
Final Boundary Constraints	Final boundary constraint violation
Parameter Constraints	Parameter constraint violation

Simply select the scalar of interest (e.g. *const_norm*) and press the **Show** button to display the graph in the *Viewer* window. The **Show in New Tab** button displays the graph in a new tab.

As mentioned above, all general settings concerning the plot style and how to arrange plots in the *Viewer* window are described in Section 2.2.

Tip: Show displays the selected scalar in the currently active plot whereas **Show in New Tab** creates a new tab in the *Viewer*.

6.12.2 Grid History Graph

The grid iteration data file (`\gismo\grid_iter.dat`) stores the iteration histories of all state and control variables as well as path constraint violations as function of time. Moreover, it contains the collocation defects or violations of multiple shooting point continuity conditions at each major grid point (*state_name.CC*).

The purpose of this graph type is to monitor how a function evolves over subsequent iterations. For all iterations (or only some, there are several selection criteria) a line plot is drawn for the grid item-pair. The line plot associated with the current iteration is highlighted with a thicker line and always drawn in red; the oldest iteration is drawn in blue.

Simply select two variables from the *X-Axis* and *Y-Axis* lists (e.g. *Time* and *R*), and press **Show** to display the graph in the *Viewer*. **Show in New Tab** displays the graph in a new tab.

Tip: Show always displays the selected items in the active plot (red-framed) of the *Viewer*.

All general settings concerning the plot style and how to arrange plots in the *Viewer* are described in Section 2.2. However, some special settings are present in the *Viewer* window, when a *Grid History Graph* is displayed (see Fig. 6.5):

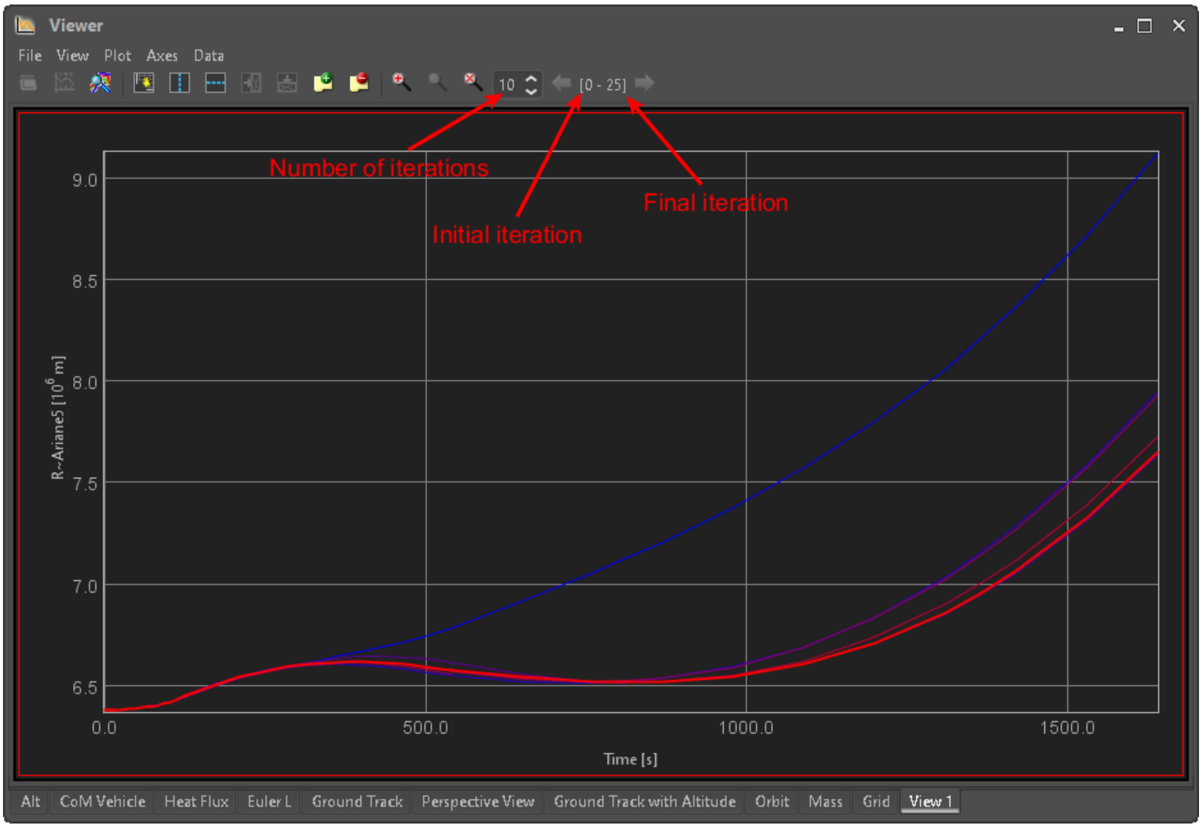

Fig. 6.5: Special settings in the Viewer

The rightmost input fields specifies the number of iterations that shall be printed (10 in Fig. 6.5). Apart from an explicit input, also the up/down icons can be used to increase/decrease the number of iterations. The icons ⇦ and ⇨ can be used to adapt the initial and final iterations to be displayed.

7 Multibody and Flexible Dynamics

This chapter gives a detailed description of the **Multibody dynamics** feature in *ASTOS* and of the required work-flow to perform simulation based on it.

In order to set up a scenario with flexible component parameters, sloshing properties, clampband devices and multibody capabilities, the **ASTOS Multibody dynamics** add-on must be installed and the **Multibody dynamics** feature must be enabled in the root node of the scenario *Modelling* tree.

The different sets of equations of motion in *ASTOS* (except for the *(Flexible) Multibody Dynamics*) consider the entire vehicle as one rigid body, i.e. as one mass with inertia and do not consider any multibody capabilities. However, there are cases where the user might want to model the vehicle as flexible or the simulation might involve more than one body.

For example the flexible contribution might be relevant when the following effects are not negligible and need to be considered:

- the effect of the fuel moving in tanks (sloshing)
- the dynamics of a satellite with (flexible) solar panels
- the bending of the ascending rocket structure
- the aero-elasticity effect, etc.

Moreover, the multibody capability alone allows the user to simulate dynamics of multiple bodies interacting with each other such as the separation of multiple payload from the adaptor and upper stage, the jettison of contiguous stages in a launcher, the separation of vehicle side boosters, etc.

The **Multibody dynamics** feature provides two different simulation approaches:

- *Linearised Flexible Dynamics* (see Section 7.1) is an upgrade of the standard *ASTOS* equations of motion that allows to include the flexibility effect of the vehicle during the simulation
- *(Flexible) Multibody Dynamics* equations of motion (see Section 7.2) allow the simulation of multibody systems where components can be assumed as rigid or flexible

Even if both approaches allow the simulation of the vehicle flexibility, there are some main differences between them:

- the *Linearised Flexible Dynamics* considers the overall flexibility of the vehicle as a whole and only in terms of frequencies and mode shapes. While the *(Flexible) Multibody Dynamics* equations of motion consider the dynamics of each single component (for example the movement of a spring-mass system).
- the *Linearised Flexible Dynamics* allows the user to consider only a specific number of modes for the system flexibility (at vehicle level) while the *(Flexible) Multibody Dynamics* assumes the modes defined at the component level (with no limit for the total vehicle modes)
- the *Linearised Flexible Dynamics* considers only the perturbation of external forces on the vehicle flexibility while the *(Flexible) Multibody Dynamics* takes into account the inertial forces too
- the flexible properties in the *Linearised Flexible Dynamics* are pre-computed and do not depend on any simulation parameters. However, some effects (e.g. the sloshing model, see Sloshing in the Model Reference book) depend on the system evolution. This dependency is directly considered by the *(Flexible) Multibody Dynamics*.

7.1 Linearised Flexible Dynamics

By using the **Multibody dynamics** feature, every component can be modeled as a flexible beam. Moreover, additional flexible elements can be added such as sloshing, clamp band or spring-damper actuators.

The *Linearised Flexible Dynamics* method augments the regular *ASTOS* dynamics model of the scenario by including a set of equations of motion (modelling the linear system flexibility) on top of the *ASTOS* equations of motion. The settings and the amount of flexible modes the user would like to include can have an effect on how computation-intensive is the simulation.

Information about the corresponding mathematical model is given in chapter Linearised Flexible Dynamics in the Multibody and Flexible Dynamics book.

Usage

Once the scenario is built including flexible elements, the following steps are required to simulate the scenario with the *Linearised Flexible Dynamics*:

1. Export: the MBS export must be performed in order to generate the external files containing the flexible properties of the system at specific Points of Interest (POIs) along the scenario mission time. Specific output nodes useful for the flexible dynamics are automatically generated by the algorithm (actuators and sensors mounting nodes, distributed and non-distributed aerodynamic nodes and sloshing mass nodes).
2. Linearization: the user must drag and drop the MBS export output folder (by default is `..\MBS`) on the `mode_shape.bat` file (in the **ASTOS Multibody dynamics** add-on folder) in order to generate the `MACmodes` files in each POI folder.

3. The user needs to enable the *Flexible Dynamics* panel in the vehicle *Default Settings* settings (*Modelling→ Dynamics Configuration→Vehicle & POIs Dynamics*) and then click on one of the **Simulate** buttons in order to start the simulation.

Note: The linearised flexible dynamics cannot be used with *(Flexible) Multibody Dynamics* equations of motion (see Section 7.2). Those equations of motion in fact already embeds such kind of flexible effect.

Configuration

The following input are needed:

- *Number of modes* specifies the total number of flexible modes (and therefore of flexible equations of motion)
- *Path* defines the location of the folder containing the POI subfolders with the `MACmodes` files. By default is `..\MBS`.
- *Scaling factor* is a numerical value used to scale the system frequencies, mode shapes and damping values.

Tip: It is highly suggested to use pair number of modes for symmetrical vehicles.

MBS export

The *MBS export* is used to generate files containing the multibody model of the scenario at specific points in time called Points Of Interest (POI).

Note: The MBS export cannot be performed if *(Flexible) Multibody Dynamics* equations of motion are selected.

During the export, *ASTOS* creates an output folder and inside it generates several subfolders for each POI (minimum every 5 seconds and maximum 15 per phase). The export makes use of a reference simulation for the trajectory and the mass consumption.

Inside each POI folder there are the essential files describing the multibody model:

- BEA files contain the properties of each flexible component present in the model
- DEF file is the **ASTOS Multibody dynamics** add-on definition file which describes the multibody model
- TXT file accommodates the correspondences between **ASTOS Multibody dynamics** add-on and *ASTOS* items

Every file name (e.g. `POI_pi_pt.DEF`) indicates the time to which it belongs in the mission scenario:

- pi indicates the phase index which starts at 01

- pt indicates the time point in a phase, where 00 stands for the initial phase time. If only maximum two data points are exported per phase the final phase time is indicated as 01
- pi and pt consist always of two integer digits

Tip: Those files can be later utilized to linearize the multibody models and thus analyse the mode shapes and the frequencies of the system. Those steps are required to include the flexible dynamic effect using the *Linearised Flexible Dynamics* panel in the vehicle *Default Settings* settings (*Modelling→ Dynamics Configuration→Vehicle & POIs Dynamics*).

The following input are needed:

- *Description* can be used to add information about the export
- *Simulation input file* specifies the `struct` file used as reference for trajectory and mass consumption
- *Output folder* contains all the POI subfolders (by default is `..\MBS`)
- *Vehicle* defines the vehicle to be exported

Note: The simulation used as reference must have exactly the same vehicle configuration and the same number and length of phases.

BEA and DEF files

The BEA and DEF files are ASCII files containing the properties of the flexible multibody system defined by the *ASTOS* scenario. Refer to the *DCAP* User Manual for more information [40].

POI_pi_pt.TXT file

The TXT file provides information to the *CON* tool regarding the **ASTOS Multibody dynamics** add-on configuration. It provides a list of all physical interested nodes.

The *ASTOS* identifier is used by the *CON* to identify the column inside the `simulation.txt` file. The information inside this file is created specifically for the definition of a linearised dynamics of the controller. The GNC engineer needs to be able to map the node index with the mode shape file in order to associate the correct mode shapes to the right node. Moreover he needs the *CNA* values at that position for the aerodynamics.

The textual file has a fixed structure:

- Number of nodes
- List of node indices
- Phase index
- Phase time
- Number of nodes related to actuator and sensor equipment
- List of **ASTOS Multibody dynamics** add-on node index and *ASTOS* identifier

- Number of nodes related to distributed aerodynamics nodes
- List of **ASTOS Multibody dynamics** add-on node index, local aerodynamic CNA coefficient and node x-axis position with respect to the global reference frame

Modes file

The `MACmodes` file it is read by the *CON* tool for controller design purpose and for the linearised flexible dynamics.

The file consists of the following global information:

- Number of frequencies
- System frequency values (in *Hz*)
- Modal mass (in *kg*)
- Modal relative damping

The remaining part of the file is divided in several blocks, one for each node, reporting the following information:

- **ASTOS Multibody dynamics** add-on node index
- Position (x,y,z) with respect to the global frame (in *m*)
- Mode slope around x-axis
- Mode slope around y-axis
- Mode slope around z-axis
- Mode shape around x-axis
- Mode shape around y-axis
- Mode shape around z-axis

The user is given the possibility to use custom flexible input. More information is given in Section 7.1.1.

7.1.1 Custom Flexible Properties

In case the implemented model do not fully satisfy the mission requirements, the user is given the possibility to use custom flexible input.

After the MBS export and the **ASTOS Multibody dynamics** add-on linearisation are performed, an output folder with POI subfolders is present. The user can now retrieves the following information for each POI:

- time (in the `TXT` file)
- nodes location with respect to the global node (in the `MACmodes` file)

Using those information the user computes the frequencies, damping values and mode shapes for each node and for every POI with any other external tool. Finally the user needs to overwrite the values in the `MACmodes` file for each POI.

The *Linearised Flexible Dynamics* algorithm extracts the information from the `MACmodes` files, using then the values provided by the user.

7.2 (Flexible) Multibody Dynamics

The *(Flexible) Multibody Dynamics* equations of motion allow the simulation of multiple bodies interacting with each other. Moreover each vehicle component can be assumed as flexible and additional elements (spring-damper actuators, sloshing, clamp band, etc) can be included in the scenario.

Due to the increased number of equations for the flexibility and/or the multibody model, the mathematical complexity of the equations of motion is much greater making therefore the entire simulation slower.

Furthermore, the system flexibility makes the natural frequencies of the entire vehicle greater than zero. Such natural frequencies are then a limit to the integration step size allowed for the simulation as explained by the Nyquist theorem [45]. For example if the smallest system natural frequency of the vehicle is 5 Hz, the integration frequency cannot go theoretically below 10 Hz (2x of the system frequency as indicated by the Nyquist theorem). However, it is a good practice to stay far away from the minimum required and simulate with a frequency of about 50 Hz.

The *(Flexible) Multibody Dynamics* equations of motion uses the open-loop Order(n) dynamic formulation based on the *DCAP* software. For more information about the equations of motion refer to the *DCAP* User Manual [40].

Usage

Once the scenario is built including flexible elements, the following steps are required to simulate the scenario with the *(Flexible) Multibody Dynamics*:

1. Click on one of the **Simulate** buttons in the *ASTOS* GUI.

Configuration

In order to make use of the flexibility and multibody capabilities, the *(Flexible) Multibody Dynamics* must be selected as equations of motion and the following setting is also required:

- *Attitude reference element* is the element of the vehicle which represents the attitude (orientation and rotation) of the vehicle. The center node of that element is used as reference.

Note: Each component with a non-zero structural mass introduces a set of 7 equations of motion (3 translational and 4 rotational with quaternions).

In order to perform a simulation using the *(Flexible) Multibody Dynamics* set of equations of motion, the following requirements must be fulfilled:

- A fixed step-size integrator must be used
- All phases must use the *(Flexible) Multibody Dynamics* equations of motion
- The global node must be connected to the bottom node of the first equipment
- The first equipment directly connected to the global node cannot be jettisoned
- The attitude reference element must be defined for every phase

Moreover the following limitations exist:

- Multi-vehicle scenarios are not supported
- Finite element models from *NASTRAN* software are not supported
- Only the gridless simulation (initial guess and connected phases) is fully supported (see Section 6.5)

Files and folders

When the simulation with *(Flexible) Multibody Dynamics* equations of motion is performed, the `*.gtp/model/MBS` folder is created with all the multibody information stored in dedicated files.

The different files specify the mechanical multibody system topology, the flexible properties, the time dependent properties (mass, CoM, inertia, etc), the symbolic multibody equations of motion (see Multibody Dynamics in the Multibody and Flexible Dynamics book.

For more information about the files format refer to the *DCAP* User Manual [40].

8 Results Inspection and Presentation

The results of a simulation, optimization or analysis can be accessed, customized and plotted using the tools in the *Results* tree (Fig. 8.1). A scenario animation can be generated by the *Astroview* tool (Section 8.3) which is part of the *ASTOS* GUI .

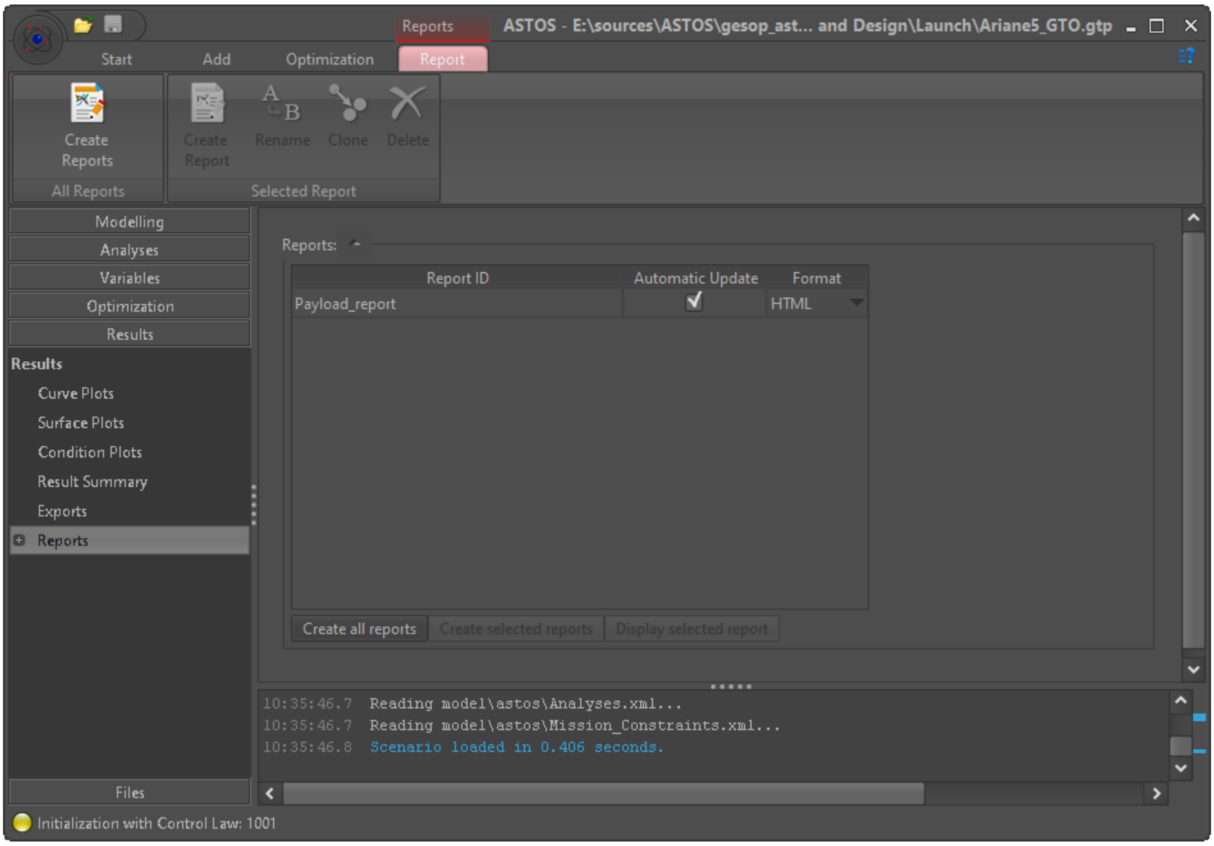

Fig. 8.1: The Results tree is shown on the left side of the window

The *Results* tree contains several nodes. Some of them are directly linked to a configuration panel while others simply define a group node for other subnodes, e.g. the *Reports* node groups all reports defined for the current scenario. In general, the following top-level nodes are available:

- *Curve Plots* (Section 8.2.1) allow the user to select the items (e.g. states, controls or auxiliary functions) to be plotted in the *Viewer*. The configuration panel facilitates the customization of the plot data, the selection of the data sources and the selection of the plot type.

- *Surface Plots* (Section 8.2.2) allow the selection of items (e.g. auxiliary functions) to be plotted as a three-dimensional shaded surface.
- *Condition Plots* (Section 8.2.3) allow the user to define certain boolean conditions (e.g. a certain angle below or above a user-defined value). The result can be plotted in the *Viewer* to visualize the fulfillment of the boolean conditions over time.
- *Result Summary* dumps a textual representation of the result data to the *Execution Log*. For further details on how to configure this feature, refer to Section 8.5.
- *Exports* serves as a group node for all defined exports (for instance, certain results can be exported as *MS Excel* files). Selecting the *Exports* node opens an overview panel, which contains a list of all defined exports. In this panel, exports can also be marked as automatic, i.e. they are created each time a simulation, optimization or analysis is completed. Additionally, (all or only selected) exports can be created manually from this overview panel. Selecting one of the subnodes opens the configuration panel for the associated export. See Section 8.6 for details.
- *Reports* serves as a grouping node for all defined reports. The report feature is explained in detail in Section 8.4. Selecting the *Reports* node opens an overview panel, which contains a list of all defined reports. In this panel, reports can be marked as automatic, i.e. they are are created each time a simulation, optimization or analysis is completed. Furthermore, the format of the report can be specified here. Additionally, (all or only selected) reports can be created manually from this overview panel. Selecting one of the subnodes opens the configuration panel for the associated report.

8.1 Types of Output and Naming Conventions

Overview of available outputs

ASTOS provides the user with different kinds of outputs:

- *Independent variable*
  Input to differential equations and all output functions. Available as independent variable is time or the equinoctial true longitude as angle.
- *States*
  Computed at each major grid node either analytically or numerically via integration of the state derivatives (right hand side).
- *Controls*
  A control is a function of the independent variable. The optimizer is free to change the value of a control inside the specified bounds at each control evaluation node.
- *Path constraints*
  Type of constraint evaluated at each constraint evaluation node except the initial and the final one.
- *Lagrange cost*
  Type of cost function evaluated at each major grid node except the initial and the final one and integrated as a function of the independent variable.

- *Auxiliary Functions*
  Additional output as function of the independent variable. It may reflect anything being neither state nor control.
- *Auxiliary Scalars*
  Phase specific constant value with a unit. Only available in Reports (see Section 8.4).
- *Auxiliary Items*
  Set of constant values of one or more Auxiliary Functions.
- *Auxiliary Events*
  An event based on an auxiliary function and the independent variable. It can either be the maximum or the minimum of an auxiliary function or a predefined value of this auxiliary function.

For further information about these terms see Chapter 11.

Output file location

Every model component, constraint or analysis may define output functions and most of them are computed during simulation and analysis of the scenario. All output functions are stored in simulation output files located in the sub-folder `integ` of the scenario gtp folder.

Naming conventions

A unique ID and description is automatically assigned to each output function and parameter, and the naming follows fixed rules in order to have a clear understanding for the user. Each ID is composed of several strings connected by symbols identifying the relationship between two adjacent strings:

- '->' means 'to' (e.g. visibility of sensor to station).
- '#' means 'in', before frame or coordinate system (e.g. in J2000).
- '@' between frame and center object (e.g. J2000 at Earth).
- ':' between object and parent (e.g. antenna1 part of vehicle1).
- '~' between output function and object (e.g. x of vehicle1).
- '&' means 'and' between two objects without hierarchy (e.g. ratio between entity1 and entity2).

In case of a coordinate system the ID is *frame@center* (e.g. J2000@Earth is the frame J2000 centered at Earth).

Auxiliary Functions

The following table gives examples for auxiliary functions. These names can be found in the *Curve Plots* node of the *Results* tree. For details about the *Curve Plots* node see Section 8.2.1.

Table 8.1: Examples of ASTOS functions with symbols

Name	Explanation
x~Vehicle1#J2000@Earth	x-position of Vehicle1 in coordinate system J2000 at Earth

Name	Explanation
x~Vehicle1#J2000@L1:Earth	x-position of Vehicle1 in coordinate system J2000 at L1 part of Earth
spin_rotation~Earth#J2000	rotation angle of prime meridian of Earth in frame J2000
z~Antenna3:Vehicle2#J2000@Groundstation2	z-position of Antenna3 part of Vehicle2 in coordinate system J2000 at Groundstation2
x~Antenna1:Vehicle1#J2000@Antenna2:Vehicle2	x-position of Antenna1 part of Vehicle1 in coordinate system J2000 at Antenna2 part of Vehicle2
prop_tank~Booster_Tank:Vehicle3	propellant mass of Booster_Tank part of Vehicle3
visibility->Groundstation2~Antenna3:Vehicle1	visibility between Antenna3 part of Vehicle1 and Groundstation2
visibility->Antenna1:Vehicle1~Antenna5:Vehicle2	visibility between Antenna1 part of Vehicle1 and Antenna5 part of Vehicle2
ratio~booster1&booster2	ratio between booster1 and booster2

Auxiliary Scalars

An Auxiliary Scalar can be any phase-specific quantity independent of the independent variable. A good example is the structural mass of a vehicle. The structural mass of a vehicle is constant within a phase, since it only changes when parts of the vehicle are jettisoned and the jettison always defines the end of one phase. Auxiliary Scalars exist for:

- geometric information of the vehicle (e.g. length, diameter etc.) and its individual parts
- structural masses of the vehicle and vehicle parts (stages, payload, fairing etc.)
- certain outputs of the link-budget analysis
- position of a Ground Station
- Risk analysis (probabilities for casualties etc.) and impact area of fragments (only in destructive reentry analysis)

These Auxiliary Scalars can be part of a report and are not visible otherwise. For further information about the reporting feature see Section 8.4.

Auxiliary Items

An *Auxiliary Item* is a set of constant values of one or more *Auxiliary Functions*. One example is the position of a user defined Ground Station or the splash down position of jettisoned boosters and stages of a launcher. The position of a Ground Station is specified by longitude, latitude and altitude. Since a Ground Station does not move the values of longitude, latitude and altitude don't change over time. Every user defined Ground Station creates an *Auxiliary Item* containing the position of this Ground Station. This *Auxiliary Item* can be visualized in a map plot. On how to add Auxiliary Items to plots see Section 2.2.1.

Note: *Auxiliary Items* can be added to plots only if the *Auxiliary Functions* referenced by this item are contained in the plot, e.g. the Auxiliary Item for the Ground Station needs a plot containing at least the Auxiliary Functions longitude and latitude of any vehicle if it is a 2D map plot and longitude, latitude and altitude if it is a 3D map plot.

Auxiliary Events

As the name suggests an *Auxiliary Event* is an event during the simulation of a scenario. In the current version of *ASTOS* the following events are available:

- maximum acceleration
- maximum dynamic pressure
- maximum heatflux
- mach equal one

These events can be visualized in plots. On how to add Events to plots see Section 2.2.1.

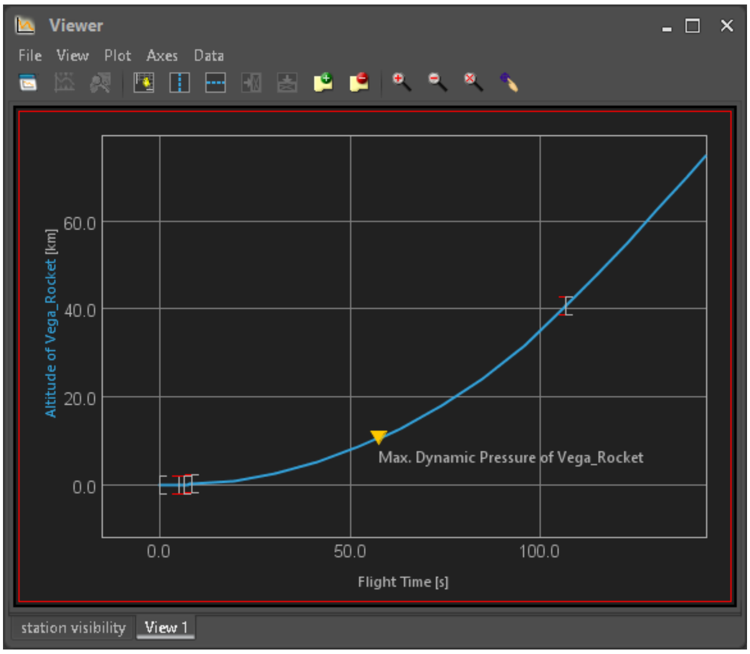

Fig. 8.2: Example of a 2D plot containing the auxiliary event max. dynamic pressure

8.2 Plots

ASTOS contains a functionality to plot numerical data and thus enables the user to easily visualize and compare the results of simulations and analyses. Data which is stored in either `*.struct`, `*.tstruct` or in text files can be plotted. The data source files are specified in the respective plots panels. All data plots are displayed within tabs inside the *Viewer* window.

This window can be opened by selecting using the *Start→Windows→Viewer* action button. The *Viewer* window is described in Section 2.2 in detail.

ASTOS distinguishes the following plot type, each of them being accessible the corresponding node in the *Results* tree:

- *Curve Plots* generate line plots. Different coordinate systems and representations are available. Refer to Section 8.2.1 for more details.
- *Surface Plots* generate three-dimensional shaded surfaces. More details are provided in Section 8.2.2.
- *Condition Plots* generates bar graph diagrams and display the result of boolean equations evaluated over the mission duration. A detailed explanation is given in Section 8.2.3.

Common plots configuration

The various plot types have a common set of configurations which is explained in the following. Each plot type contains a *Data Source* panel to load data input files from different simulations or even `.txt` files created with other applications. This facilitates the graphical comparison of the simulations of different scenarios or the comparison of the simulations of one scenario for different parameter values. By default, after a simulation run, two simulation data sets are available: the *Initial Guess* simulation data set and the *Simulation* data set. In the *Data source* drop-down list it is possible to select the desired data source.
The button **Add Data Source** in the *Data Source* panel allows the user to insert additional data source files containing e.g. one or more of the following:

- simulation data from simulation runs for various scenario configurations and settings
- simulation data from different scenarios
- simulation data obtained from another software or source to be plotted together with the scenario simulation data

The **Add Data Source** button opens a window in which the desired `.struct` or `.txt` file can be specified. The choice has to be confirmed with the **Open** button. In case the data file is a `.struct` file, it automatically appears in the *Data Source* drop-down list. If it is a `.txt` file, a template has to be created to drive the data import from the file. For an explanation of how to define this template file, please see section Section 8.2.4.
Apart from the default data sources *Simulation* and *Initial Guess*, other entries in the *Data source* list can be removed by selecting them and clicking on the **Remove Data Source** button.

The content of the selected data source item in the *Data Source* list is shown as a data tree structure in the *Simulation Data* panel. In this tree structure the identifiers of the quantities can be double clicked or dragged and dropped into an axis field of the *Plot* panel in order to define plots. In the *Plot* pane several curves may be defined and all of them will be plotted together in one plot. All plots are accommodated in the *Viewer* window and can be arranged in separated tabs (see Section 2.2).
The *Simulation Data* section lists all data items, i. e. independent variable, states, controls, path constraints, Lagrange cost and auxiliary functions present in the selected *Data Source*.

The following filters can be applied to the data items list in order to simplify the browsing of data items:

- *Element* specifies a vehicle part i.e. an element in the e.g. *Actuators* or *Components* branches of the *Vehicles Parts & Properties* sub-node of the *Modelling* tree
- *Object* specifies a vehicle or ground station i.e. an element in the *Vehicles & POIs Definition* sub-node of the *Modelling* tree
- *Frame* specifies the coordinate frame (does not include the coordinate center definition) for the representation of the data item
- *Center* specifies the center of the coordinate frame which can be the CoM of any of the following: celestial body, vehicle or ground station
- User-defined string filter

Once a filter setting has been selected in one of the filters, the data in the *Simulation Data* section is filtered accordingly immediately. When one filter setting is applied, all data items fulfilling this filter condition and also all items for which this filter condition is not defined, are shown.

8.2.1 Curve Plots

The *Curve Plots* are used to visualize and evaluate graphically results generated by a simulation or analysis.

Plot

Which parameters to plot and how they define the axes of different curves is specified in the *Plot* panel. First a *Plot type* should be chosen from the available options:

- *2D-Plot*, Section 8.2.1.1
- *3D-Plot*, Section 8.2.1.2
- *Map-Plot*, Section 8.2.1.3
- *3D Map-Plot*, Section 8.2.1.4
- *Satellite plot*, Section 8.2.1.5

If *Map-Plot* or *3D Map-Plot* is selected, another drop-down list called *Projection* appears to select the projection of the map loaded as a background into the current diagram. The available projections are:

- *Mercator*: Lines of constant bearing (rhumb lines) are straight
- *Miller Cylindrical*: Intended to resemble Mercator while also displaying the poles
- *Equidistant Cylindrical*: Distances along meridians are conserved
- *Stereographic*: Projection which preserves angles but not distances and areas
- *Orthographic*: Perspective projection with perspective point at infinity

- *Azimuthal Equal Area*: Preserves distances but not angles
- *Azimuthal Equidistant*: Distances from center are conserved
- *Gnomonic*: Great circles map to straight lines

The projection selected for each plot type is saved and reloaded when selecting the same plot type.

A table to edit and show the current plot setup is displayed in the panel. It defines what curves to draw and which parameters are associated with each axis. It is a tree-like table with its rows grouped in curves, and each curve is drawn as a line graph in a plot. The first row of a curve is coloured in light grey and can be collapsed to hide their axis rows. A curve row has the following column naming:

- *Name:* Curve name
- *Source/Function:* Data source of the parameters associated with their axes
- *Units:* Not used

An axis rows has a slightly different column meaning:

- *Name:* Axis name
- *Source/Function:* Function name of the parameter
- *Units:* Units of the parameter

To fill a curve with parameters they can be dragged from the *Simulation data* panel and dropped onto the curves table or double clicked directly. The filling behaviour is different depending on whether a curve, an axis or nothing in the table is selected. If an axis is selected or the parameter is dropped over an axis, it is inserted (if possible) exactly in that place.
If a curve is selected it is automatically inserted in the first empty axis which allows a valid configuration for that curve. If it can't be inserted, a new curve is generated and auto-filled if possible.
If nothing is selected, all curves are search from top to bottom for an empty axis where the parameter can be inserted. If no one is found, a new curve is generated to insert it.
Double clicking parameters in the *Simulation data* list is a fast way to define a plot with only a few mouse clicks. Once the first curve is defined, all the following automatically generated curves have their domain axes pre-filled with the same parameters as the previous one.

All parameters from a specific curve must be from the same source, but it is possible to create two or more curves with different sources and draw them in the same plot. The axis units also impose restrictions on which parameters can be inserted. This limitations are described with more details in the corresponding *Plot type* sections.

The name cell in axis rows can be selected to show a selection list which allows to swap it with an other axis. Of course the units of these swapping axis parameters must be compatible with the new configuration. Also when axes with the same name exist in other curves they must swap too.

To edit the curves table the following control buttons are also available:

- All: Deletes all curves
- : Deletes the selected curve with their axes or removes the parameter from a selected axis
- **Show**: Opens the *Viewer* window and visualizes the curves defined in the *Plot* table
- **Show in New Tab**: The same as **Show** but a new tab for the plot is generated in the *Viewer* window

8.2.1.1 2D-Plot

The *2D-Plot* type can be used to plot functions of one independent variable (2D functions). One data item has to be selected for each axis pair x and y respectively (e.g. time for the x-axis and altitude for the y-axis). An example of a typical *2D-Plot* is shown in Fig. 8.3.

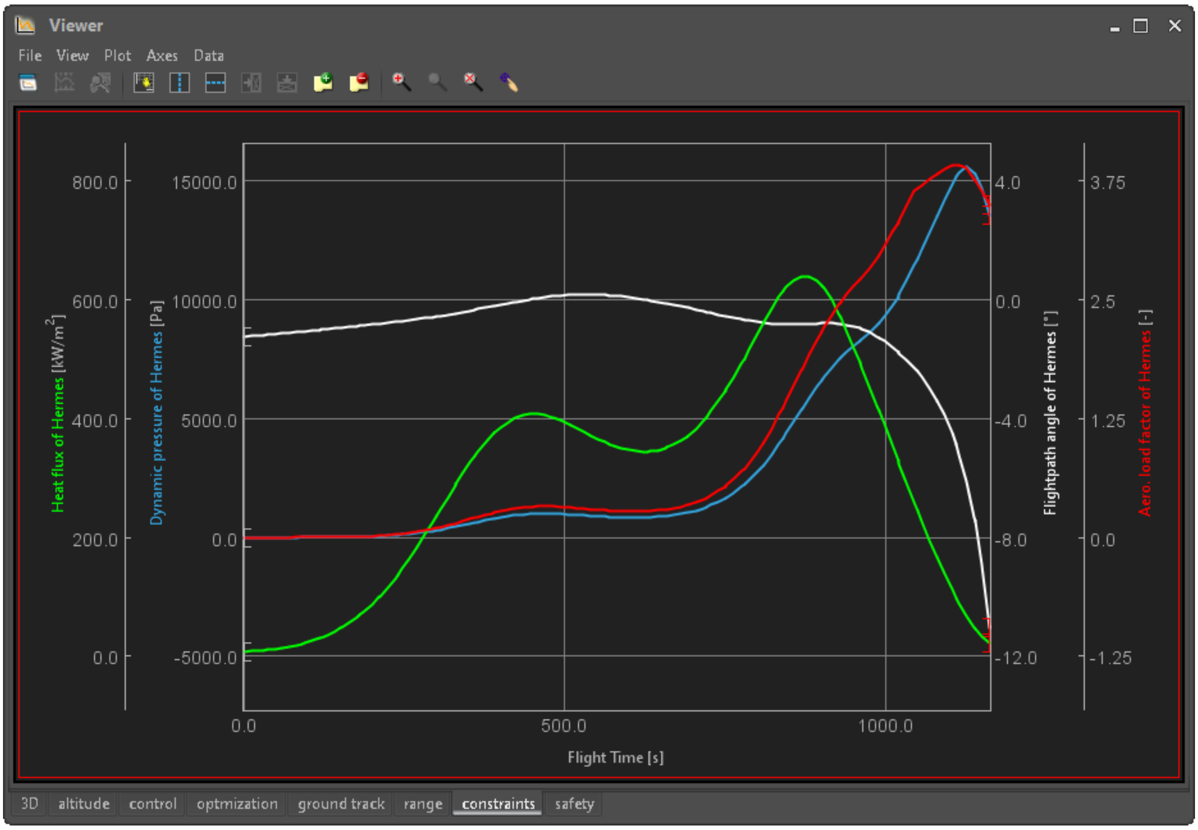

Fig. 8.3: A 2D plot with four y-axes

A *2D-Plot* in ASTOS can accommodate an arbitrary number of plots. The following restrictions apply, however. On the x-axis only quantities with compatible units (e.g. time) can be plotted. On the y-axis up to four axes (Y1, Y2, Y3 and Y4) with independent units and value ranges are supported simultaneously. Plotted curves are grouped by their y-axis unit and each group is assigned to its own y-axis in the plot.

There are two ways to create a *2D-Plot*. The first two quantities selected (double-click with the mouse) in the *Simulation data* tree are assigned to the x- and y-axis respectively of a newly created curve plot. Any further quantities selected in the *Simulation data* tree are inserted into the y-axis of a new curve, while the x-axis quantity is adopted from the previous curve plot definition.

Pick Data

A *2D-Plot* has a data picking functionality which is activated by pressing the button (see Fig. 8.3). Simply right-click close to the function graph and a point on it is captured. A box next to the captured point shows the numerical X and Y coordinates of the point and their units.

> **Tip:** The numerical values are also saved in the clipboard and by pressing 'Ctrl+V' they can be pasted in your favourite program (*Word*, *Notepad*, *Excel*, etc.).

Controls

The following control buttons are available in the *2D-Plot*:

- : Zoom in. This shall be used in order to return to a previous zoom level.
- : Zoom out. This shall be used in order to return to a previous zoom level.
- : Sets the zoom level to fit all data in the plot view
- : Enables the data picking functionality

The *2D-Plot* supports the following interactive zoom-in functionality:

- By pressing and holding the right mouse button and by moving the mouse the user can span a box. Once the button is released the zoom of the box is applied to the plot.
- The user can zoom in along one axis of interest. This however will also affect the zoom level along the remaining axes in the plot. The mouse cursor should be placed between the axis label and the axis line and then at the position of the one end of the desired zoom interval. While keeping the right mouse button pressed, the user shall drag the mouse cursor in the direction of the other end of the desired zoom interval. A double sided arrow with short vertical lines at its tips is shown while dragging in order to indicate the selected interval to zoom in. Once the right mouse button is released the resulting zoom level is applied. The button shall be used in order to restore a previous zoom level.

The *2D-Plot* supports the following interactive pan functionality:

- By pressing and holding the left mouse button and by moving the mouse the user can pan the plot view.
- The user can pan the view along one axis of interest without affecting the view along the remaining axes in the plot. For this purpose the mouse cursor should be placed between the axis label and the axis line and Then keeping the left mouse button pressed, the user shall drag the mouse cursor in the desired pan direction. Panning along the x-axis and along the y-axis in case that there is only one y-axis in the plot is performed smoothly. In case of panning along a y-axis while there are more than one y-axes in the plot, the

panning movement is performed step-wise in order to retain the grid lines which are shared by the y-axes.

Axis Labels

The axis label text is generated from the description text of the plotted quantities and their units. In case of need the text can be modified in the *Axes* menu where the desired axis sub-menu (i.e. *X*, *Y1*, *Y2*, *Y3* and *Y4*) contains the *Caption Text ...* option allowing the user to either restore the *Default* axis label text or to provide a *Custom* one. In case of a custom label text the colour of all axis label texts in the current plot is set to the default text color in the *ASTOS* GUI. In case the *Default* setting is selected, only the axis label text is restored but not the colours. The axes label colours can be activated and deactivated for all axes simultaneously by toggling the *Use Colored Captions* setting in the *Axes* menu. The *Use Colored Captions* is greyed out in case for at least one axis label the *Custom* option is active.

8.2.1.2 3D-Plot

This plot type is an extension of the *2D-Plot*. It consists of one x-axis, one y-axis and a z axis. Only the x and y axis units have to be compatible for all curves, the z axis parameters share the same plot axis even when they have incompatible units. Four is the maximum number of curves to print simultaneously in a *3D-Plot*.

When auto-filling a *3D-Plot* the first three selected parameter are inserted sequentially into the x, y and z axis of a newly created curve. All following selected parameters are inserted directly into the z axis of a new curve and the x and y axes are copied from the previous one.

A *3D-Plot* has the following control buttons:

- : Isometric preset view 1
- : Isometric preset view 2

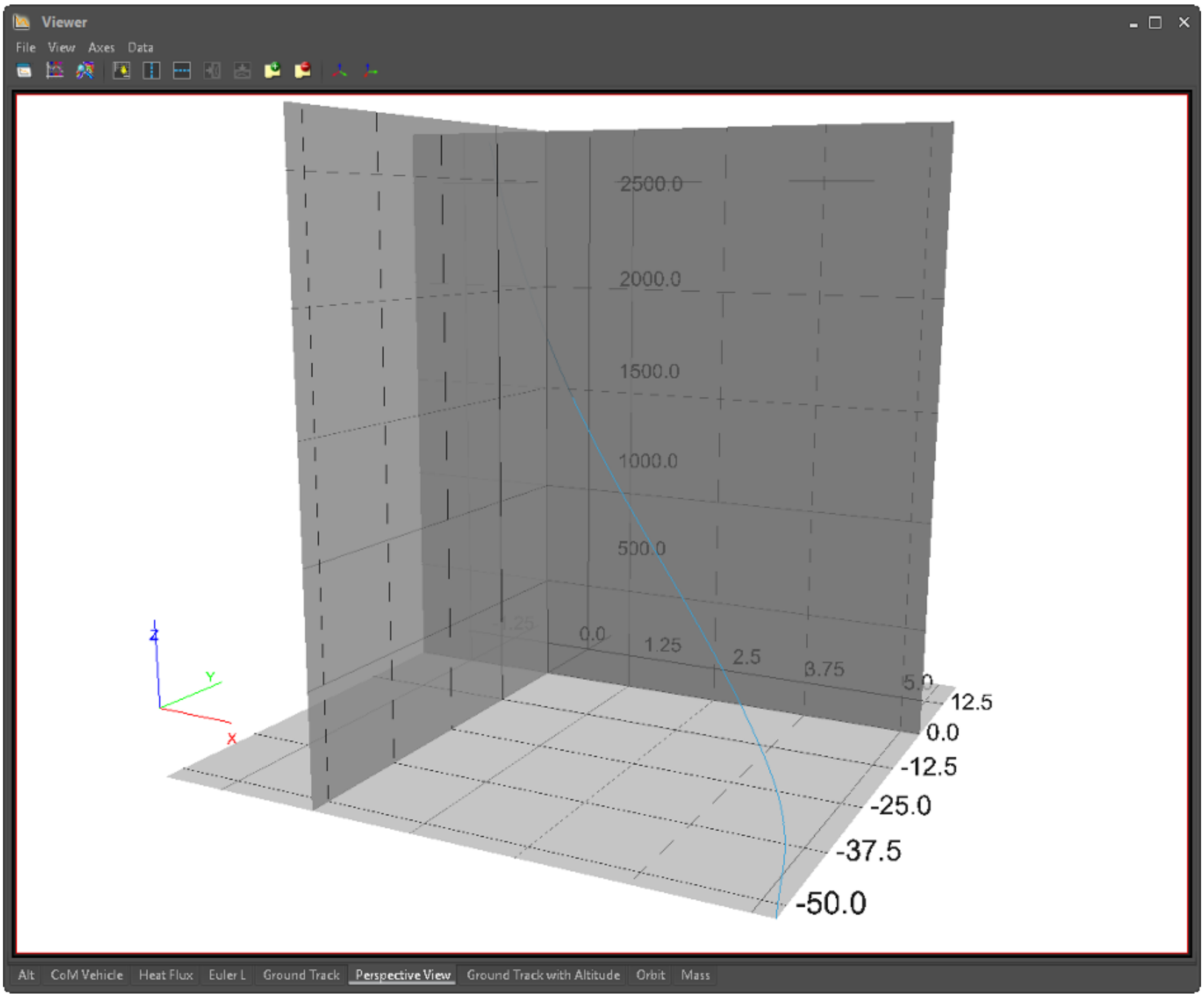

Fig. 8.4: A 3D Plot

8.2.1.3 Map-Plot

This plot is used to visualize scalar values in a positional coordinate system. It consists of a longitude and latitude axis with the third parameter used as a scalar function visualized by line colouring. For the longitude and latitude axis, only positional data having a unit compatible with degrees can be selected. For the line colour, the unit is not a limiting factor.

When auto-filling a *Map-Plot* the first two selected parameter are inserted sequentially into the longitude and latitude axes if their units are compatible with degrees. If a parameters unit is not in degrees it gets inserted into the *Line Color* axis of the curve. If the curve has no empty *Line Color* slot, a new curve is created, the parameter inserted and the longitude and latitude axes copied from the previous one.

The **Add Defaults** button allows to add a new curve with its coordinate axes latitude and longitude already filled with the position parameters of a vehicle defined in the model.

This plot type also offers the possibility to choose from several background maps. As default, the surface of the Earth is used for the background, but also population or air traffic density maps are available. Additionally, this plot is able to visualize GIS (geographic information system) data. GIS data can be provided as shape files (`*.shp`, `*.dbf`, `*.shx`) or as TIFF images (`*.tif`, `*.tiff`, `*.geotiff`). GIS data is organized in layers. These layers can be managed by clicking on the icon in the *Viewer* toolbar.

A *Map-Plot* has the following control buttons:

- : Zoom in (Alternatively the right mouse button can be used to draw a box and zoom into it)
- : Zoom out
- : Sets the zoom level to fit all data in the plot view
- : Opens the

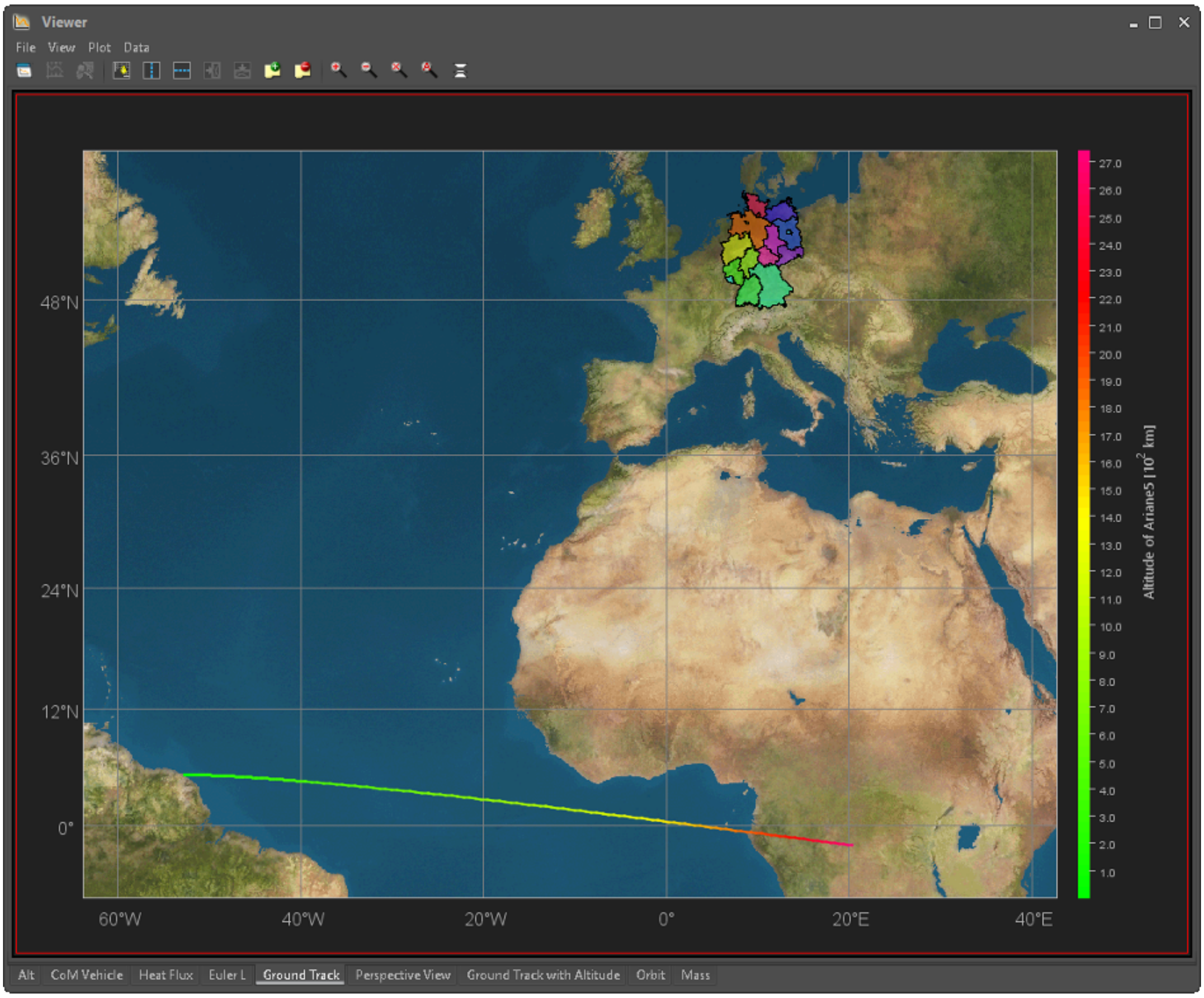

Fig. 8.5: A Map Plot visualizing the altitude as line color

GIS Layers Manager

With the *GIS Layers Manager* images and layers can be loaded and configured to be displayed as overlays on the map.

Available layers :

- *Images*
- *Point shapes*
- *Line shapes*
- *Polygon shapes*

Depending on the layer type the following configuration options are shown on the parameter panel.

Image parameters:

- *Visible*: Sets if the layer should be visible or hidden
- *Image style*: Allows to choose between *Grayscale image* and *RGB image*
- *Color bands*: The color bands for each gray or color channels can be selected

Shape parameters:

- *Visible*: Sets if the layer should be visible or hidden
- *Shape*: For points the displayed symbol can be chosen to be circle, square, triangle, cross and X
- *Line color*: Sets the color of the line shapes and the border from points and polygons
- *Fill color*: Sets the fill color for polygons and for points if they are displayed as circles, triangles or squares
- *Traverse attribute*: If Traverse is selected as fill color, a shape attribute can be selected to be used as color identifier. This will assign the same color for all shapes which have the same value for the selected attribute
- *Opacity*: Sets the opacity of the fill color
- *Point size*: Sets the point size for point shapes
- *Line width*: Sets the line width for line shapes
- *Label attribute*: Selects the attribute to be displayed as a label next to each shape

8.2.1.4 3D Map-Plot

This plot type extends the *Map-Plot* by an altitude axis. For this new axis, only data with a unit compatible with meter can be used. This plot type also offers the possibility to change the map background as well as to visualize GIS data.

The **Add Defaults** button allows to add a new curve with its coordinate and altitude axes already filled with the position and altitude parameters from a vehicle defined in the model.

A *3D Map-Plot* has the following control buttons:

- : Zoom in (Alternatively the right mouse button can be used to draw a box and zoom into it)
- : Zoom out
- : Sets the zoom level to fit all data in the plot view
- : Opens the
- : Rises the altitude range
- : Lowers the altitude range
- : Rotates the map 90 degree clockwise
- : Rotates the map 90 degree counter-clockwise

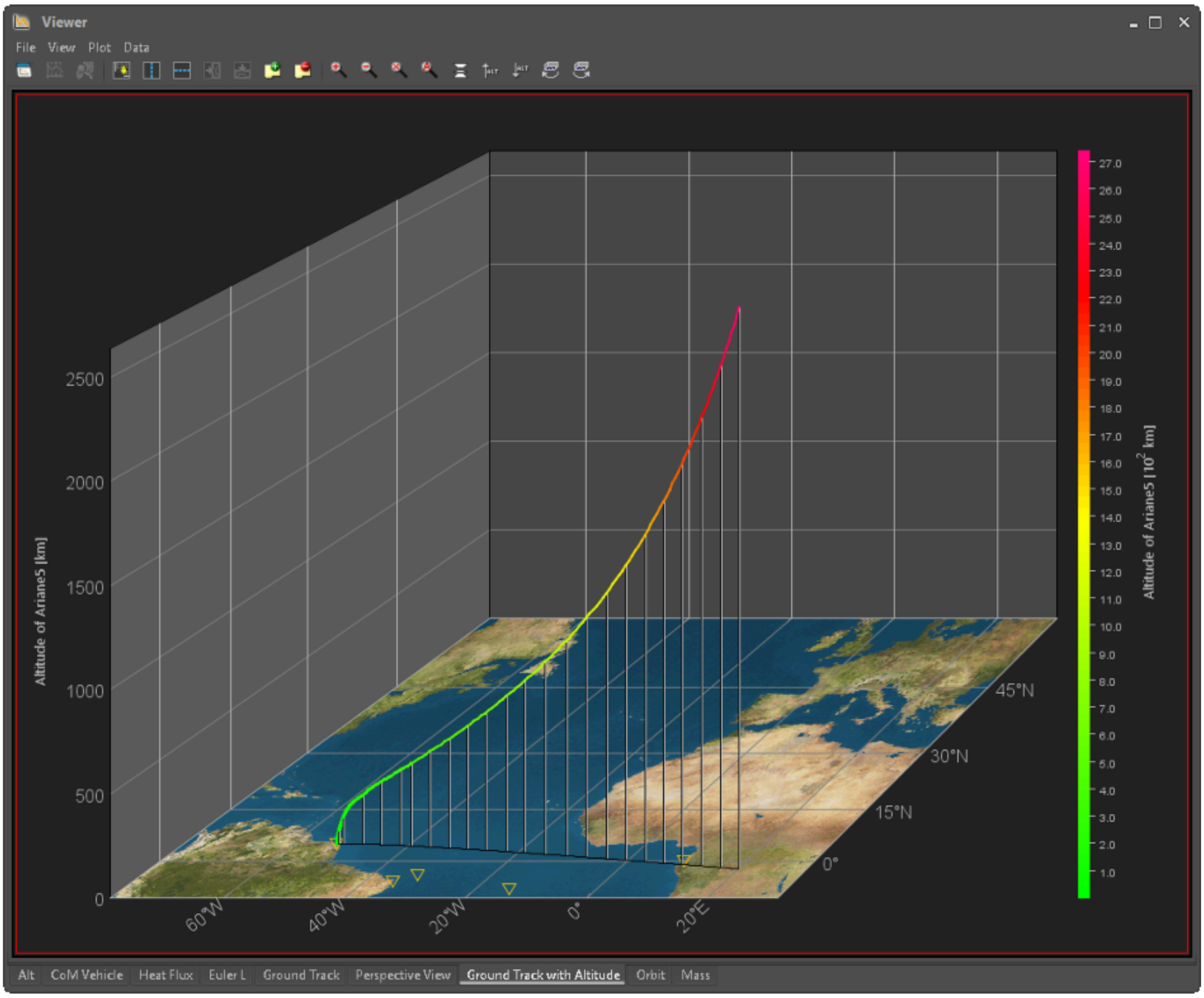

Fig. 8.6: A 3D Map Plot visualizing the mach number as line colour

8.2.1.5 Satellite Plot

Satellite Plot is a special form of *3D Map-Plot* where the Earth is visualized as a sphere instead of a flat map. Trajectories are drawn around this sphere and can be inspected from different views by rotating the earth. More info can be found in the Section 8.2.1.4 chapter.

A *Satellite Plot* has the following control buttons:

- ALT: Rises the altitude range
- ALT: Lowers the altitude range
- : Opens the

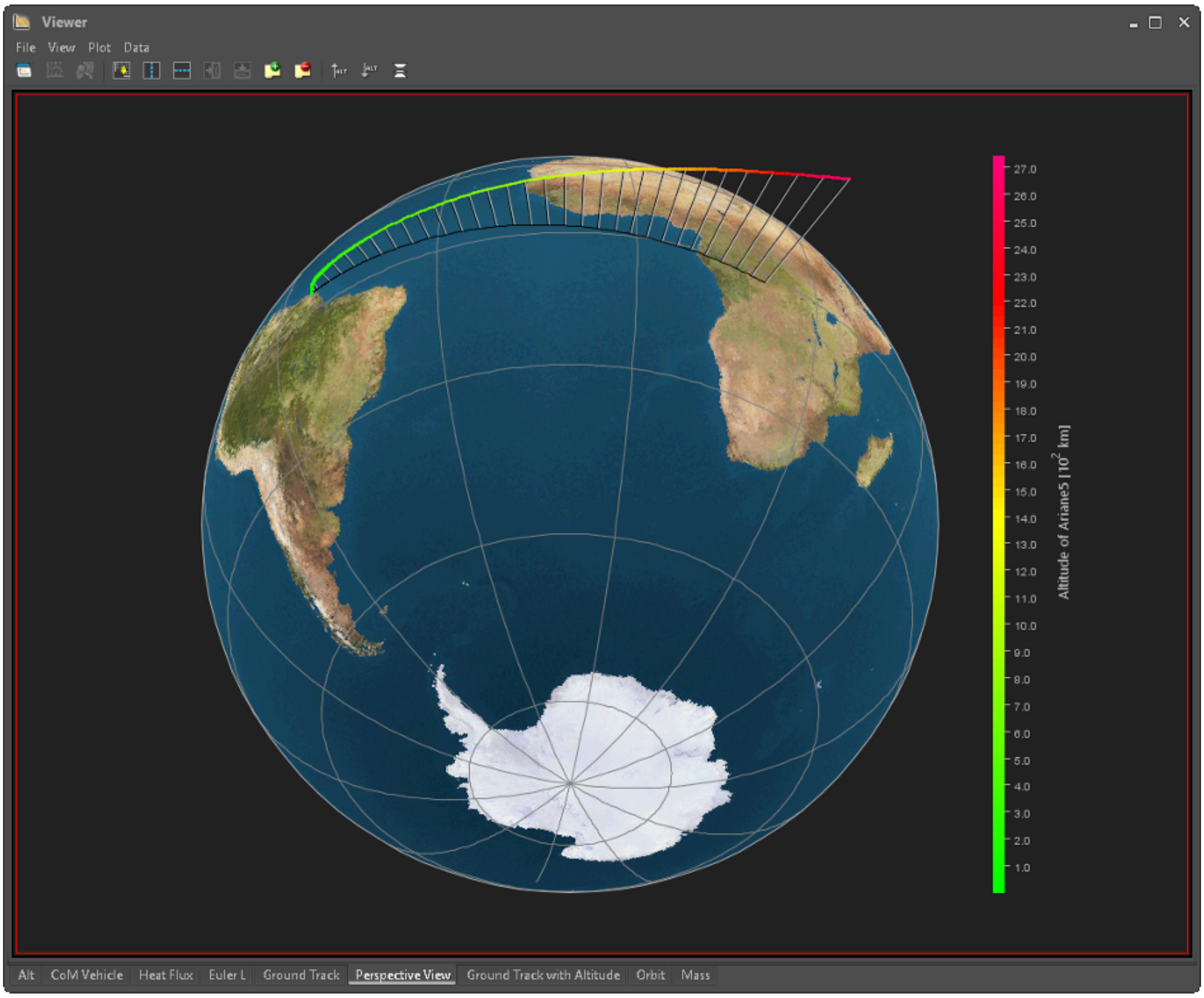

Fig. 8.7: A Satellite Plot visualizing the mach number as line colour

8.2.2 Surface Plots

Surface Plots are 3-D surface plots shaded proportionally to the surface z values. The z values have to be a function of the x and y values. A surface plot example is shown below.

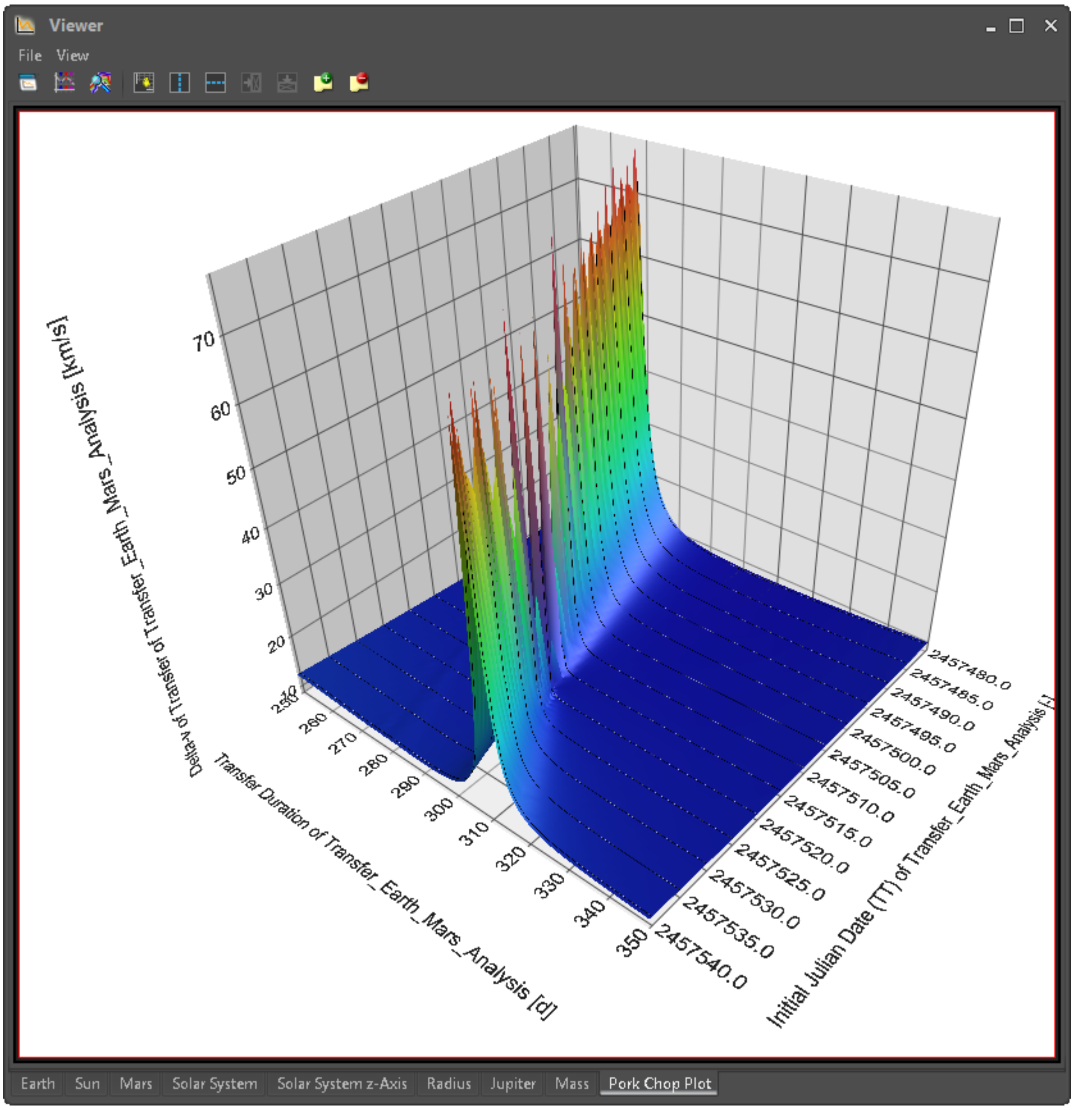

Fig. 8.8: An example of a surface plot

8.2.3 Condition Plots

A condition plot - also called a schedule diagram - shows the result of a boolean expression along the mission duration. In *ASTOS* at each stored simulation output node this user-defined boolean expression is evaluated. Boolean true values are presented as a green and boolean false values - as a red slice of a horizontal condition bar as shown in Fig. 8.9. The color of the bar may also be grey presumably if the boolean expression is not well formed. Several condition bars can be placed in a single condition plot. The user-defined boolean expression may consist of simulation output function values, constants and operators (arithmetic and logical).

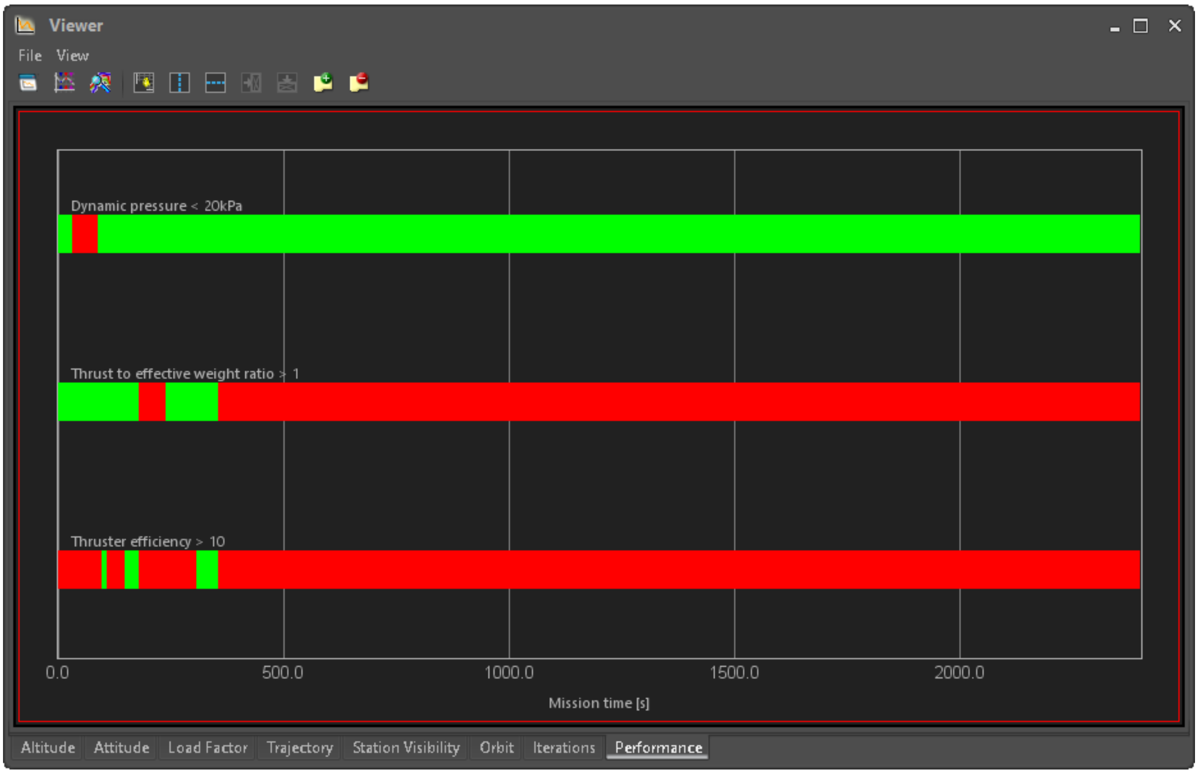

Fig. 8.9: Example condition plot

Note: Colour transitions between True and False slices will be painted in the middle between the two adjacent output nodes. This becomes important in case of sparse grids.

Creating a condtion plot

The condition plot settings panel is shown in Fig. 8.10.
Similar to the curve plots (see Section 8.2.1) the data source for the data to be plotted can be selected from the *Data Source* drop-down menu. The available output functions of the simulation data are listed again in a tree on the left of the condition plot panel in the *Simulation Data* pane. In the condition plot only one data source may be used within a single expression.

In order to add a new boolean expression a new entry has to be added to the table in the *Conditions* pane. Each row in this table represents one bar in the condition plot. The bar will be labeled with the text provided in the *Description* column. Once an entry in the table has been created and selected, the definition of the boolean expression can begin. New terms and operators (called *items* below)can be added to the boolean expression by either double-clicking an entry in the *Simulation Data* tree or by pressing one of the operator buttons or the **Value** button (this inserts a numerical constant) in the *Current Condition* pane. New items will be added at the end of the expression unless the cursor is placed somewhere else by clicking with the mouse into the expression. Single items or the whole expression can be deleted using the two delete buttons on the right of the operator buttons. Obvious errors in the expression are indicated by expression items displayed in red colour.

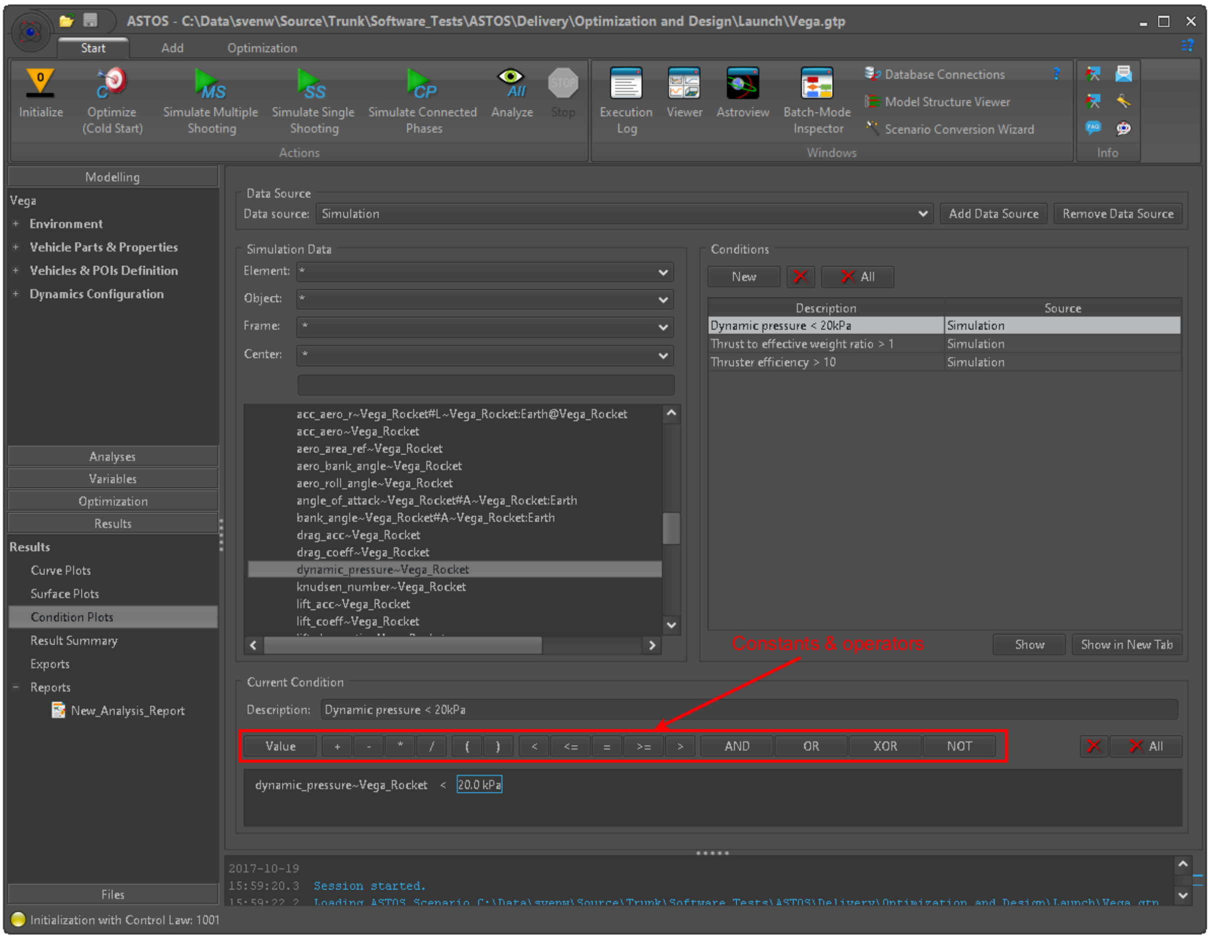

Fig. 8.10: Condition Plots panel

The boolean expression terms and operators are evaluated according to the classical operator precedence: operators are evaluated from left to right except for:

- arithmetic operator before comparisons before logical operators
- * and / before + and -
- NOT before AND, OR and XOR

Please use brackets if a different operator precedence is desired.

If a constant value is inserted into the logical expression, also a corresponding unit has to be specified. This is required to make the constant value compatible to the output functions in the simulation data which consist of a value and a unit as well. Unit conversions and compatibility checks are automatically performed internally. The unit has to be inserted as a string whose style has to be in line with the one used in any unit field in the *ASTOS* GUI (e.g. Kilo-Meter or Mega-Joule/Kilogram are valid units). Section 2.10.3 gives an overview on the correct specification of units and the available base units and prefixes.

8.2.4 Plotting External Data

In case a text file is used as a data source, a `*.tstruct` file is needed to drive the import. When a `*.txt` file is selected a dialog is displayed in which the import parameters for file can be specified. It is possible to define the number of header lines and the type of delimiters used to separate the data columns. A short preview of the data columns is shown at the bottom of the import dialog. Selecting one column inside the preview section enables the possibility to define an action for this column. The selected column can be either ignored during the data import, it can be interpreted as a phase index or it can be imported as an auxiliary function. In this case, a name, a description and a unit needs to be specified. Clicking the **OK** button opens a file dialog used for saving the imported data as a `*.tstruct` file in a user defined location.

Note: The template file contains the relative path (if applicable) of the `.txt` file and the settings about columns, not the data themselves; both files (the `.txt` and the `.tstruct`) are necessary to handle the source data.

8.3 Animations

The simulation and analysis results can be visualized via the built-in animation tool called *Astroview*. *Astroview* shows the 3D space scenario with a set of default celestial bodies of the solar system. If specified in the scenario, ephemerides and orientation of the celestial bodies are taken from the scenario settings, otherwise a simple Keplerian motion and a constant rotation rate about a fixed axis is applied.

Tip: If the *ASTOS* closes without a notification when the *Vehicle Builder* or the *Astroview* panel is opened, please refer to the Troubleshooting chapter, sub-section **GUI closes without notification** in the Software Installation Guide book for a solution.

The visual representation of a spacecraft is defined by the user in the *Vehicle Structure* (see Section 4.4.1).

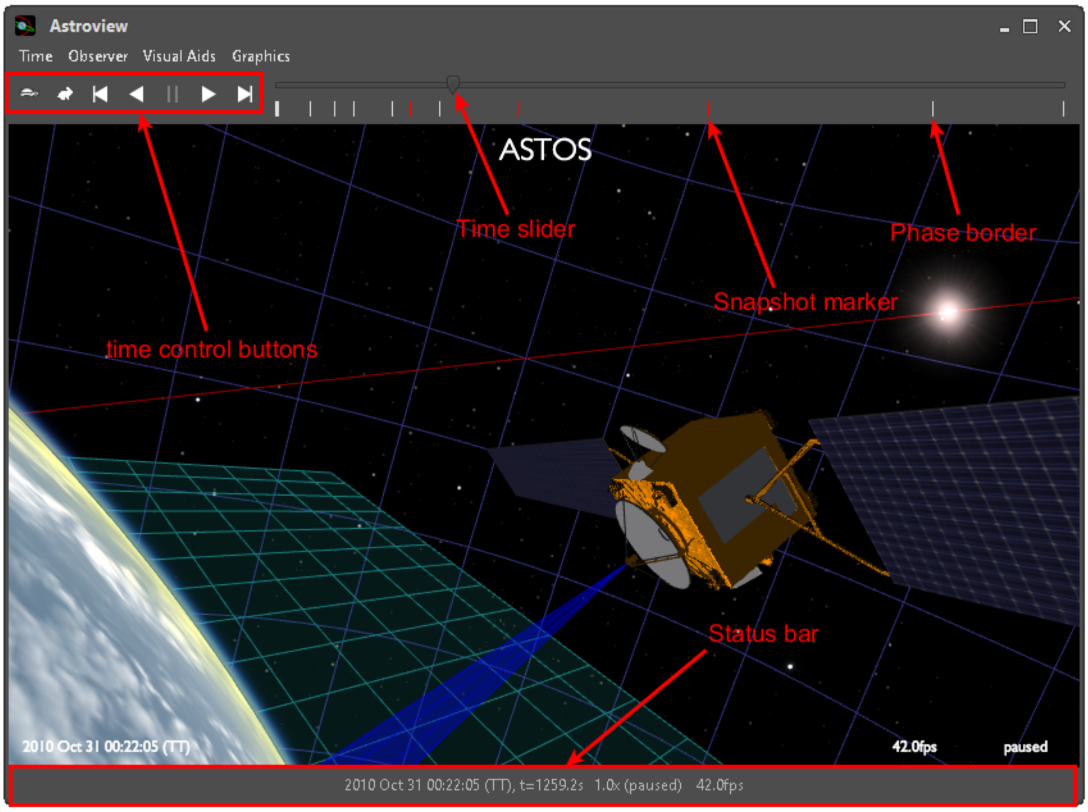

Fig. 8.11: The Astroview window

Speed control

The speed of the *Astroview* animation can be controlled using the *Time* menu, the turtle and rabbit icons on the left of the *Astroview* toolbar or using keyboard shortcuts. Pressing the *F* key or clicking on the rabbit will play the animation faster, while pressing the *S* key or clicking the turtle will slow down the animation. The animation can be paused or resumed by pressing the *space* bar. Pressing the *R* key inverts the direction of the animation. Pressing the *Shift+G* key allows the user to insert a precise time in the animation (*Time→ Go to...* menu item). The current animation speed as well as the current simulation time (UTC) are shown in the status bar.

Camera control

The position of the camera (observer) can be changed using the mouse. Keeping the left mouse button pressed while moving the mouse will let the camera orbit around the selected target. A target can be selected via the *Observer→ Set Target...* menu item. Keeping the right mouse button pressed while moving will let the camera rotate around its own position. With the scroll wheel the camera is moved towards or away from the target. Pressing *Ctrl+G* or via the menu item *Observer→ Go to Target* The field of view of the camera (the observer view) can be adjusted as well via the menu (*Observer→ Set Field of View...*). The frame definition in the *Observer* menu defines how the camera moves during the animation: either following the body fixed axis of the vehicle or the inertial axis.

Adjusting the visual quality

Via the *Graphics* menu several characteristics that influence the realism of the animation can be controlled. Switching off some of these visual features is recommended in case of poor frame rates caused by less powerful graphics cards or very complex scenarios.

- *Normal Map* - Adds a relief to celestial body surfaces
- *Shadows* - Let vehicle parts cast shadow on other surfaces
- *Reflections* - Adds reflections to glossy surfaces that then mirror the scene
- *Atmosphere* - Shows the Rayleigh scattering of the atmosphere
- *Cloud Layer* - Displays a static cloud layer
- *Ambient Light* - Adds ambient light to the scene

Adding more ambient light reduces the realism of the scene but provides a better vision in case of solar eclipses.

Visual aids and display overlays

The human eye view shown by *Astroview* can be overlaid with additional information. Currently the following visual aids are provided for all vehicles:

- Body frame axes
- Air path frame axes
- Inertial frame axes (J2000)
- Inertial velocity vector (J2000)
- Nadir vectors
- Trajectories; the entire trajectory of a vehicle can be visualized or only a part of it, e.g. start to current time; the line style (e.g. colour) can be customized.
- Sensor frustums; it is defined by the optional cone angle setting and not by the antenna characteristics; the colour, transparency and range can be customized.
- Target vectors
- Coverage; the covered area is identified by green dots, the not covered area by red dots.
- Celestial grid (equatorial)
- Ecliptic
- Equatorial plane
- Planetographic grid (longitude-latitude grid)
- Description of the currently displayed phase
- Icons and labels for vehicles, POIs and ground stations

Visual aids can be controlled via the corresponding menu *Visual Aids*. The frames follow the software convention: X is red, Y is green and Z is blue.

Video recording

It is possible to record the animations shown by *Astroview*. The recording can be started via the *Observer*→ *Record Video...* menu item. Then a target video file must be specified and afterwards the desired video codec. Depending on the selected codec further settings can be made. Video compression codecs must be installed on the system to use this feature. The drop down list shows just compatible codecs.

Note: After the target video file is selected, a dialog appears in which codec and further settings can be selected. On some systems this dialog opens but stays in the background.

In order to improve the quality of the recorded animation, it is possible to define key frames (snapshots). Taking a snapshot (*Observer→ Add Snapshot* or *Ctrl+S*) saves the current target, location and orientation of the observer. When the playback mode is activated (*Time→ Playback*, location and orientation of the observer is interpolated from those stored in the snapshots. This allows a smooth camera movement in recorded videos. Each stored snapshot is indicated by a red line below the time slider. The stored snapshots can be removed via *Observer→ Remove All Snapshots*.

8.4 Reports

ASTOS provides a reporting system which is able to create reports based on data originating from a simulation, an optimization or from analyses. All available reports can be accessed through the *Results* tree which is located on the left side of the main window. Directly clicking on the *Reports* node will bring up the report overview panel shown in Fig. 8.12.

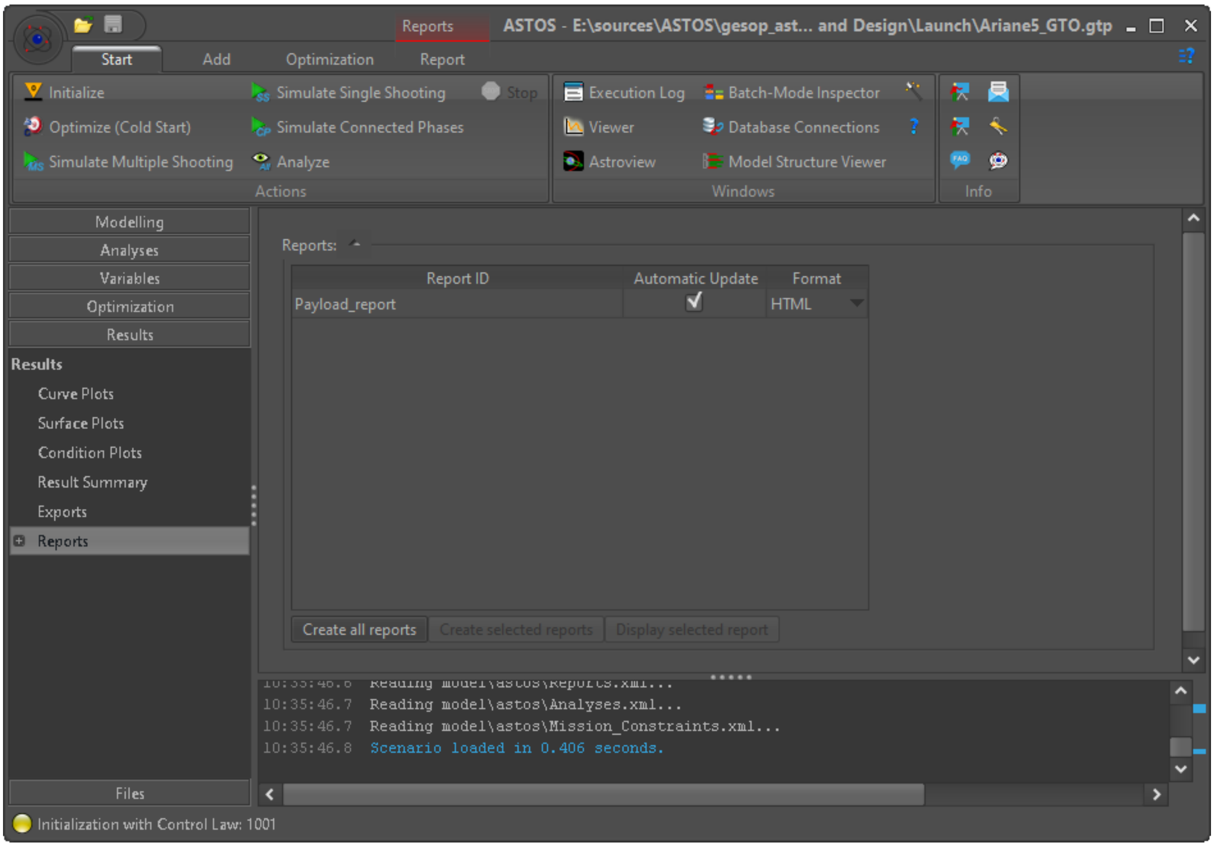

Fig. 8.12: A list of available reports is shown in case the Reports node is selected

The overview panel consists of a table containing all available reports. In this table, the following settings can be made:

- *Automatic Update* flag can be enabled, which forces *ASTOS* to recreate the report as soon as the input data has changed, e.g. after a simulation has been performed.
- The report *Format* can be selected from the *Format* drop-down list. The following output file formats are available:
  - *Word*
  - *HTML*
  - *Text*
- Additionally, the overview panel provides the possibility to manually create all reports (**Create all reports** button) or only the selected reports (**Create selected reports** button). In case only one report is currently selected, this report can be directly displayed in a separate window using the **Display selected report** button.

The content of a report can be edited by selecting the corresponding report node located below the *Reports* node. This will bring up the report panel shown in Fig. 8.13.

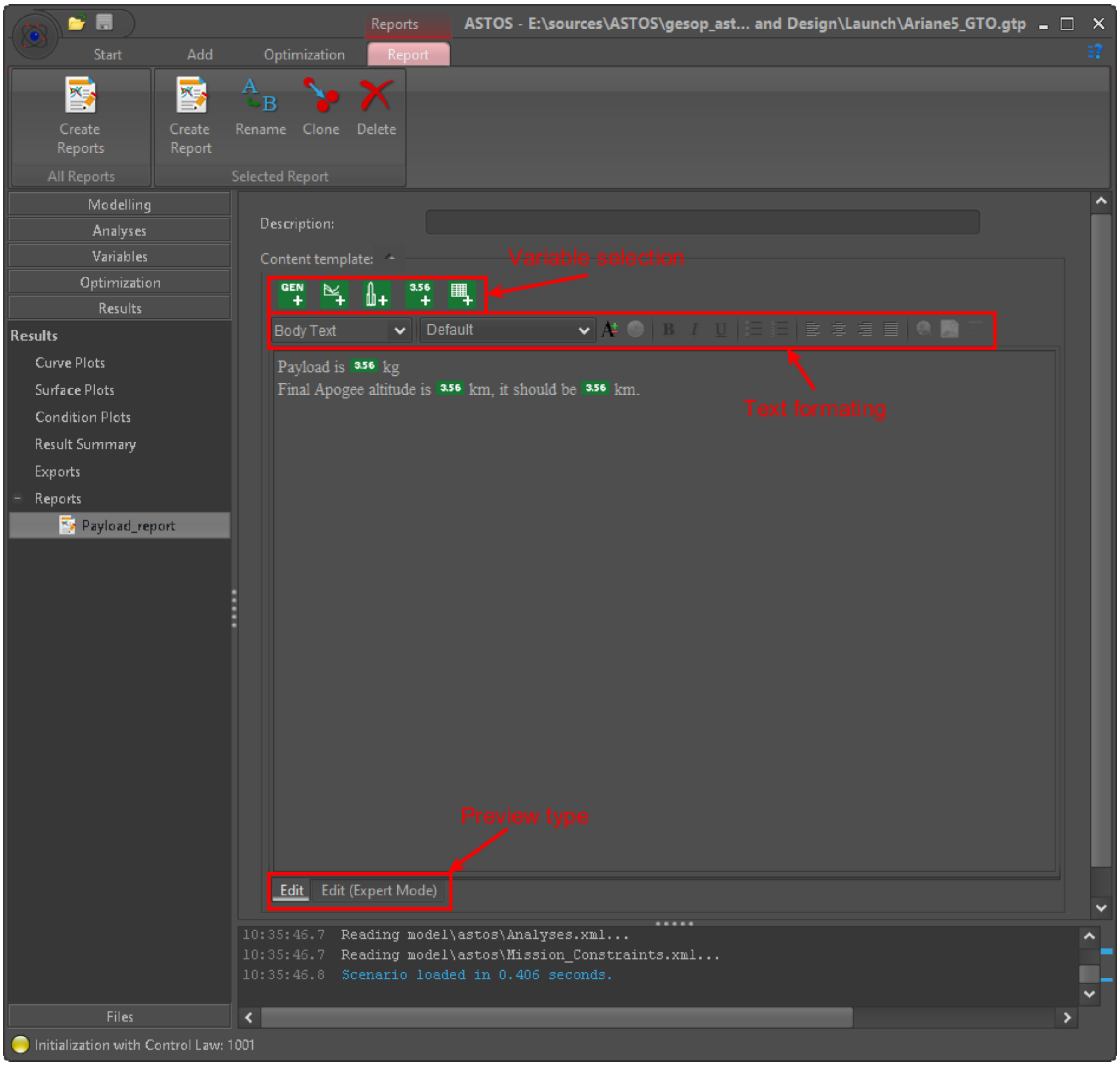

Fig. 8.13: The report creation panel

In this panel, the template for the report is created. A report template consists of text and special wildcards. Each time the report is created, all wildcards are replaced by the corresponding data. The template is defined by a HTML like structure. Because it is not required to deal with the HTML form of the template in detail, a preview of the final report is normally shown in the report panel. This preview is enabled as soon as the **Edit** tab below the editing area is selected. Some users may need to have direct access to the HTML template of the report. The raw HTML template is accessible through the **Edit (Expert Mode)** tab. In this mode, all changes must be performed by hand because all actions available in the toolbar above the editing area are not accessible.

The toolbar is split into two parts. The lower part contains actions related to the appearance of the report, like font type and font style. The upper part contains all actions used to insert wildcards into the report template. There are five categories of wildcards. For each category, a dedicated button is available in the toolbar. Moving from left to right, the following wildcard categories are available:

- The *General* category contains wildcards used to insert general information about the scenario, e.g. the scenario name.
- The *Plot* category is used to insert all kind of plots into the report.
- The *Vehicle plot* category is used to insert images of available vehicles into the report.
- The *Value* category is used to insert scalar data values into the report.
- The *Table* category is used to insert data tables into the report.

Clicking on one of these wildcard actions will bring up one of the dialogs described below. Each dialog is used to parameterize the desired wildcard.

General

In case a *General* wildcard should be inserted into the report template, the following dialog is shown. It mainly consists of a drop-down list containing all available wildcard types for this category.

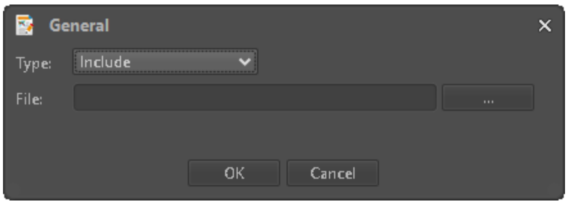

Fig. 8.14: The dialog used for adding wildcards

The following wildcards are available in the *General* category:

- *Include* provides the possibility to add content defined in a separate text file into the report. The path to the text file containing the additional content needs to be defined in the *File* text field. Pressing the **...** button opens a file dialog used to select the desired file. The content of the text file is completely inserted at the location of the wildcard as soon as the report is created.
- *Report_Description* inserts the description of the report as defined in the *Description* field.
- *Report_Name* inserts the name of the report.
- *Report_Time* inserts the creation time of the report.
- *Scenario_Description* inserts the description of the scenario. The scenario description can be set by selecting the root node of the *Modelling* tree.
- *Scenario_Name* inserts the name of the scenario.
- *Scenario_Path* inserts the full path to the scenario folder.
- *Source_Time* inserts the newest modification date of all used data sources.
- *Sources* inserts a comma-separated list of all used data sources.
- *User_Name* inserts the user name of the account which was used to create the report.
- *Version* inserts the *ASTOS* version.

Plot

The *Plot* wildcard category deals with adding plots to a report. Before a plot can be added to a report, it must be created using the *Curve Plots* panel (see Section 8.2.1), the *Condition Plots* panel (see Section 8.2.3) and the *Viewer* window (see Section 2.2). Some of the analyses create their own gavc file. In this case, it is not necessary to create the plots manually.

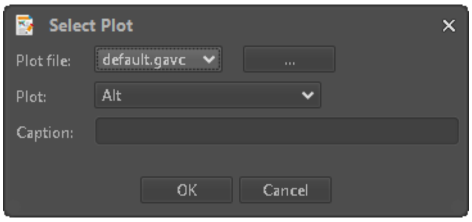

Fig. 8.15: The dialog used for adding a plot

In order to add a plot, the gavc file containing the plot definition needs to be selected from the *Plot file* drop-down list. Afterwards, the name of the plot can be selected from the *Plot* list. This list only contains the names of plots available in the current gavc file. Finally, the caption of the plot can be set. This caption is printed below the image added to the final report. After pressing the **OK** button, a second dialog opens. In this new dialog, the size of the resulting image can be defined. The size can be either set to the original size or to a user-defined size.

Vehicle plot

The *Vehicle plot* category provides the possibility to insert images of vehicles into the report. The dialog used to define the parameters for this type of wildcard is shown below.

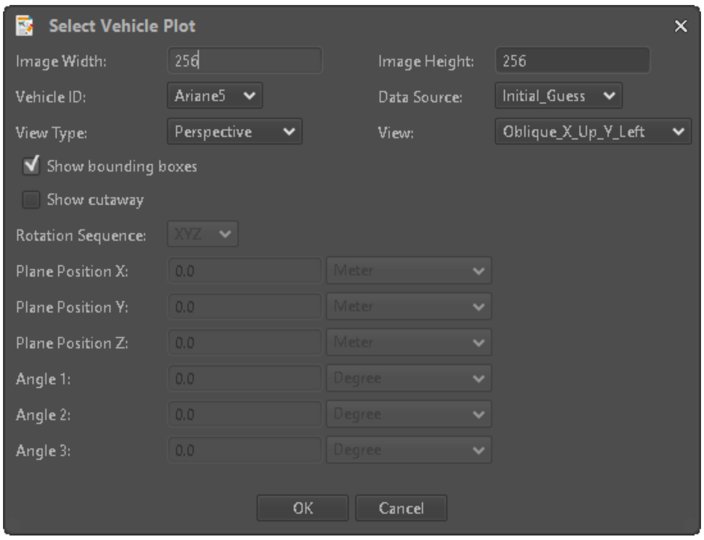

Fig. 8.16: The specialized dialog used for adding vehicle plots

The following parameters need to be set:

- *Image Width* and *Image Height*
- *Vehicle ID*

- *Data Source*. Positional and attitude data of vehicle elements can be read from two different data sources. Selecting *Initial_Guess* will use the information stored in the `ini_guess.struct` file. In case *Simulation* is selected, the information is taken from the `simulation.struct` file.
- *View Type* defines the projection used for the generated image. Possible view types are *Perspective* and *Orthographic*.
- *View*. The viewing direction. This parameter defines the orientation of the axes in the generated image.
- *Show bounding boxes*. This flag controls if bounding boxes are visible in the generated image or not.
- *Show cutaway*. This flag enables or disables a cutaway view of the vehicle. In case the cutaway view is enabled, the interior of the selected vehicle becomes visible in the generated image. Enabling the cutaway view also enables the settings for the cutting plane which are located below the *Show cutaway* checkbox. The cutting plane is defined by its position and orientation.

Value

Wildcards for scalar data values can be added by using the *Value* category button. Pressing this button opens the dialog shown in Fig. 8.17

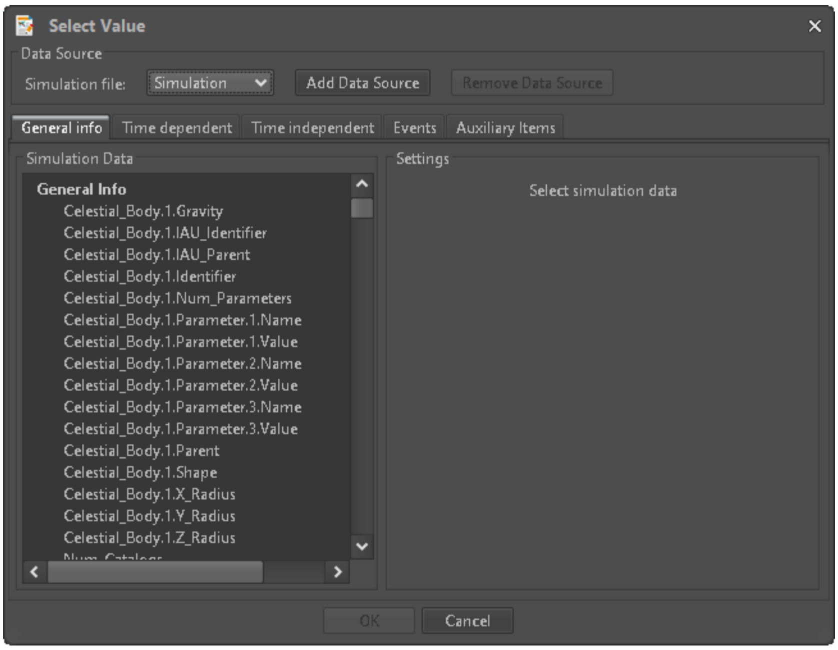

Fig. 8.17: The dialog used for adding scalar data

The upper part of the dialog is used to select the data source from which the scalar data should be taken. The data source is selected from the *Simulation file* drop-down list. It is also possible to add or remove data sources using the **Add Data Source** and **Remove Data Source** buttons.
Below the data source selection area, all available scalar values are accessible. Scalar data values are separated into 4 groups. The *General info* group contains values which are related

to the scenario itself, like number of phases or the phase names. The second group, *Time dependent*, contains scalar data which varies depending on time. The third group, *Time independent*, provides scalar values which are constant during a phase. The final group contains scalar data values related to events.

In order to add a scalar value to the report, the desired group tab needs to be selected first. Afterwards, the required data value can be selected from the tree. As soon as a data value is selected, some parameters are shown in the area on the right of the tree. The number and type of parameters depends on the data group and the selected data value itself. In general, the following parameters are available:

- *Phase index* defines the phase from which the value should be taken.
- *Unit* defines the unit in which the data value should be printed. Printing the unit is optional and needs to be enabled using the corresponding checkbox.
- *Pattern* defines which kind of data value is used. Possible values are:
    - *Initial*, the initial value at the beginning of the selected phase.
    - *Final*, the final value at the end of the selected phase.
    - *Min*, the minimum value within the selected phase.
    - *Max*, the maximum value within the selected phase.
    - *unscaled*, the unscaled value of a constraint.
    - *lower*, the lower limit of an inequality constraint.
    - *upper*, the upper limit of an inequality constraint.
    - *reference*, the limit of an equality constraint.
    - *cost*, the unscaled cost function value.
    - *rated*, if not equal zero, a percentage deviation computed by the result summary.
    - *None*, this data value has no special meaning.
- *Number of digits* defines the number of digits used to print floating point values. It is also possible to use the scientific notation by enabling the corresponding checkbox.
- *Value type* defines the data type used for events. Selecting *Event time* will print the time of the event occurrence to the report. In case *Data value* is selected, the data value of the event is printed.

Table

The *Table* wildcard category is used to define data tables. Pressing the corresponding button in the toolbar opens the following dialog.

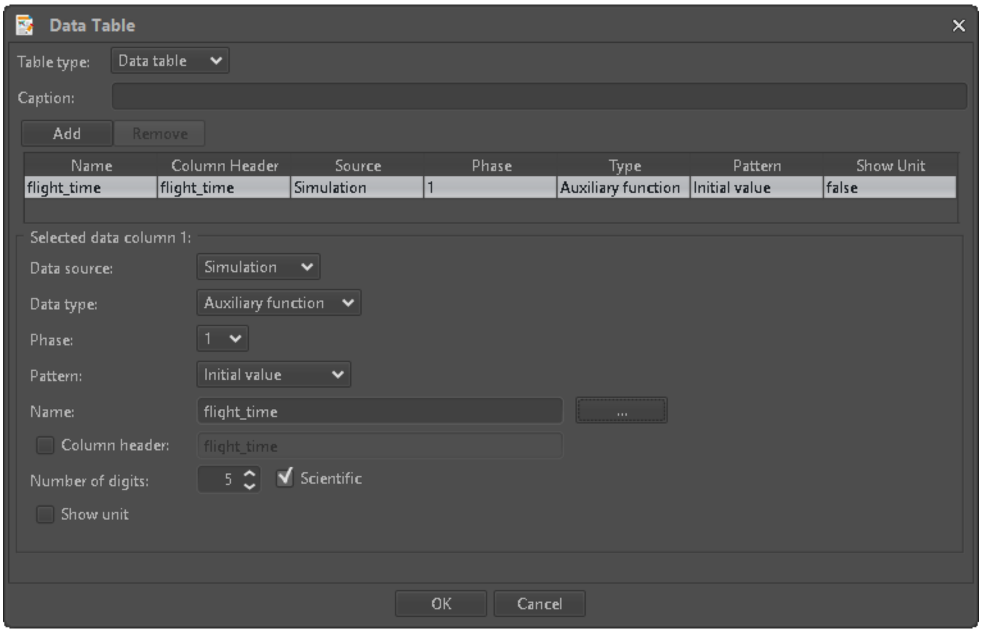

Fig. 8.18: The dialog used for adding data tables

The dialog consists of several areas. The upper part contains settings for the table itself. Below this area, there is a table area in which the columns of the final table can be defined. Each row in the table area relates to one column in the final table. Columns are added or removed by using the **Add** or the **Remove** button. Selecting one of the rows inside the table area, will bring up the configuration area for the corresponding column. This area is located below the table area. In the configuration area, the data source and type for the corresponding column is defined. The *Data source* drop down list provides the following data sources:

- *Initial guess* uses data from the initial guess stored in the `ini_guess.struct` file.
- *Simulation* uses data from the simulation or optimization stored in the `simulation.struct` file.
- *Custom...* opens a file dialog from which any `.struct` file can be selected.

After having defined the data source, the data type, e.g. auxiliary function, needs to be selected. Additionally, the name of the data needs to be defined. The name can be inserted directly in the *Name* field or it can be selected from a list which is shown as soon as the **...** is pressed. The *Name* field supports the * wildcard. The reason for this wildcard is explained a bit later.
In general, two possible table types can be selected from the *Table type* drop down list:

- The *Data table* is a generic table used to group similar data based on data names using the * wildcard into a comprehensive table. For example, in order to print the altitude for all available vehicles, one column can be defined using altitude~* as the data name. The final table will contain two columns, Item and altitude~*. In the column labeled with Item, all detected replacements for the * wildcard are listed. The altitude~* column will contain the altitude data related to the full data name consisting of the name defined for the column without the wild card, in this example altitude~, and the replacement associated to the table row which is printed in the Item column. In case no replacement can be found for a name containing a * wildcard, N/A is printed in the table cell.

- The *Event table* behaves a bit different than the *Data table* and has therefore a slightly different dialog. This dialog is shown in Fig. 8.19.

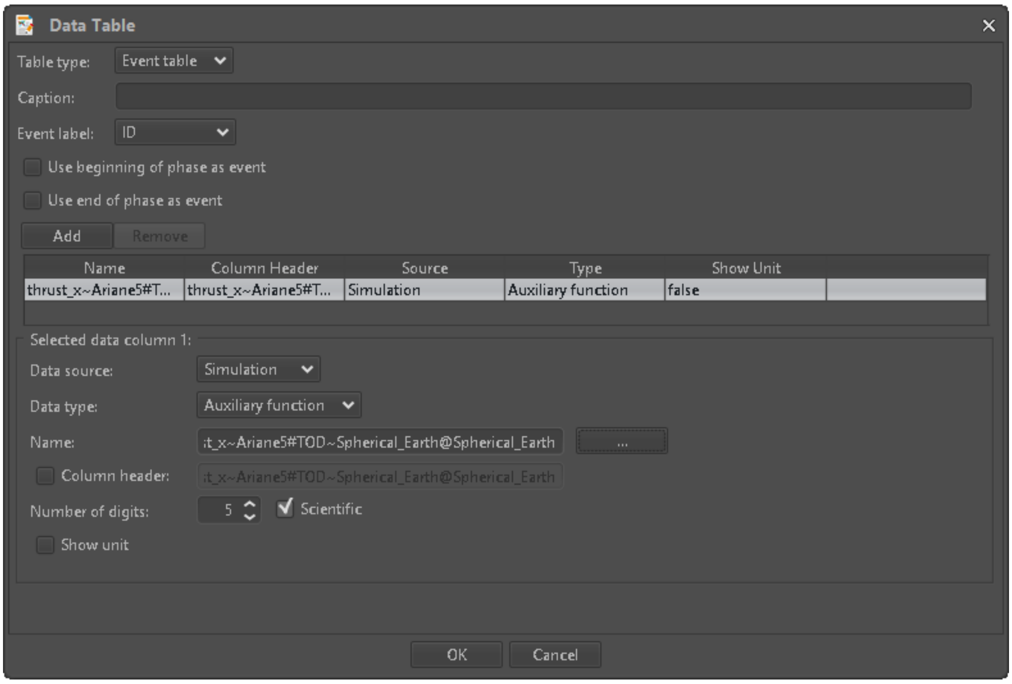

Fig. 8.19: The dialog used for adding event tables

The task of this table is to group data based on the occurrence of certain events. For each available event, one row will be created in the final table. It is possible also to consider the beginning or the end of a phase as an event by activating the corresponding checkbox. Most of the settings for this table type is equal to the settings of the data table. The main difference is, that the *Name* field inside the column definition area does not support the * wildcard. Here, the full data name must be used. Again, a list of available names is accessible through the **...**.

- The *Data timeseries table* makes it possible to extract time depend data from a *struct* file. Most of the settings for this table type are identical to the settings of the *Data table*. The main difference is that the *Data source*, *Phase(s)*, *Number of digits* and the *Show unit* settings are not individually definable for each column. The user can also specify a step size (in time) for the data points if needed. This step size does not have necesserily to match the step size in the data source. If the step sizes in the settings and in the data source do not match, the data points will be interpolated linearly if needed.

> **Tip:** Please be aware that in case of interpolation, errors can occur and the data points can therefore also vary because of this.

  - *Data source* sets the struct file which shall be used as a data source (same functionality as in *Data table*)

- *Phase(s)* provides a selection possibility between a single phase or all phases
- *Number of digits* provides the same functionality as in *Data table*
- *Show unit* provides the same functionality as in *Data table*
- *Use step size* is an optional setting for the step size for the displayed data points.
- *Name* provides the same functionality as in *Data table*, but * wildcard is not allowed
- *Column header* provides the same functionality as in *Data table*

Wildcard right-click menu

All wildcards, which are inserted in the template can be right-clicked to perform further actions on them.

Creating a report

After the report template is fully defined, the report can be created either through the report overview panel or by right-clicking on the report node below the *Reports* node.

8.4.1 Scenario Summary

The scenario summary is a different kind of report. The scenario summary is a HTML file created by clicking *Scenario*→ *Summary*. The layout of this type of report is fixed. It contains the following information:

- Cost Function
- Initial Position
- Model Description
- Dynamic System and Constraints
- Model Properties
- Environment Properties

Enforced constraints are marked as bold text in the summary. Depending on the existence of simulation data, this information is either taken from the `TOPS` file or from the `Mission_Constraints.xml` file. Consequently - as long as no simulation was executed - the `Is_Enforced` flags from the `Mission_Constraints.xml` file are used. This is necessary since (in some cases) the constraint information of the `TOPS` file may differ from the one in `Mission_Constraints.xml`.

8.5 Result Summary

Introduction

The *Result Summary* panel in the *Results* tree is used to control the layout of the result summary printout. This is a textual representation of the result data included in the `simulation.struct` file.

The output in printed in the *Execution Log* and it is divided into two parts. The first one holds a problem description and all the scalar information divided by phase: cost function, phase times, parameter and constraint values. Moreover, all the final boundary constraints of the last phase are summarized in the `PERFORMANCE INFO` area.
The second part holds the time history output of the selected items. It is only printed, if at least one item is selected via the *Item Selection* list of the *Result Summary* panel.

Tip: a reduced time history of selected functions (beginning and end of each phase) can be produced with a customizable report using *Table type*: *Event table*.

The **Summary** button performs two actions:

- The listed data is displayed in the *Execution Log* window
- A text file `result_summary.dat` is saved in the scenario root directory.

Note: The *Result Summary* should not be confused with the Scenario summary produced with the ribbon *Start*→ *Info*→ *Scenario Summary* command.

Item Selection

The *Item Selection* section includes the same list of groups as in *Curve Plots* of *Results* tree; some of them are always present: *Independent Variable*, *States* and *Auxiliary Functions*; some of them are present according to the problem characteristics: *Controls*, *Path Constraints* and *Lagrange Cost*.
To select an item just click on it, a second click deselects it. Moreover, it is possible to use the two buttons to respectively select or deselect all the items.

Page Layout

The *Page Layout* section contains all the necessary settings to customize the layout of the *Result Summary* printout.

- *Columns*: Set the number of characters per line (default 78).
- *Rows*: Set the number of rows per page in the `Result_Summary.dat` file (default 0, to avoid the indication of the page number).
- *Gap*: The gap (in characters) between each function column (default 1).
- *Digits*: The number of digits of the number format (default 10).
- *Offset*: The offset which indicates how many data sets to skip for the printout. An offset of zero therefore means that all data gets printed. A very large offset, e.g. 1000, prints only the initial and final time of each phase.

Besides the selected printing time, some predefined events are included: "Mach number equal = 1.0", "Max. dynamic pressure", "Max. heat flux", "Max. acceleration".

The *Result Summary* can be produced also in background, see Section 10.4 for more details.

Example

The following text shows an extract of a converged example printout in the *ASTOS Execution Log*.

```
******************************************************************************
*                                                                            *
* Result Summary - Scalar Information                 Apr 8, 2015 5:00:21 PM *
* Ariane 5 to GTO max payload                                                *
*                                                                            *
******************************************************************************

* COST FUNCTIONS *************************************************************
                                                           Nominal       Final
  Cost is Negative Total Payload Mass of Ariane5                   -9236.87949

* PERFORMANCE INFO ***********************************************************
                                             Desired    Achieved %-Deviation
  Apoapsis_Altitude of Ariane5               35800.0  35799.9975       7E-06
  Periapsis_Argument of Ariane5                178.0  177.999999     1.7E-07
  Orbit_Inclination of Ariane5                   7.0           7     6.9E-07
  Periapsis_Altitude of Ariane5                200.0  199.999997     1.1E-06
  Norm. residual fuel of stage PROP_TANK_L9      0.0  -2.336E-11

* SCALAR QUANTITIES OF PHASE 1 ***********************************************
  Description: Vertical Lift off
  Phase Times (initial + duration = final): 0.0 + 5 = 5 s

  Scalars                                                       Value      Unit
    Launch Altitude constraint of Ariane5                          17         m
    Launch Declination constraint of Ariane5                   5.2434         °
    Launch Longitude constraint of Ariane5                   -52.7686         °
    Launch North Velocity constraint of Ariane5            -4.261E-15       m/s
    Launch Radial Velocity constraint of Ariane5           -3.264E-14       m/s
    Launch Relative East Velocity constraint of Ariane5     1.137E-13       m/s
    Delta of Initial Propellant Mass of Booster_Left_Ta      232728.0        kg
    Delta of Initial Propellant Mass of Booster_Right_T      232728.0        kg
    Delta of Initial Propellant Mass of Stage1_Tank of       145000.0        kg
    Normalized Residual Propellant Mass of Booster_Left    224405.835        kg
    Normalized Residual Propellant Mass of Booster_Righ    224405.835        kg
    Normalized Residual Propellant Mass of Stage1_Tank         144125        kg
    Ignition Time of Booster_Left_Prop_Sys of Ariane5 o    -2.697E-25         s
    Ignition Time of Booster_Right_Prop_Sys of Ariane5     -2.335E-25         s
    Ignition Time of Stage1_Prop_Sys of Ariane5             2.008E-27         s
    structural mass of Ariane5                             92086.8794        kg
```

Payload mass of Ariane5	9236.87949	kg
structural mass of Booster_Left of Ariane5	33000.0	kg
structural mass of Booster_Right of Ariane5	33000.0	kg
structural mass of Fairing of Ariane5	1750.0	kg
structural mass of Payload of Ariane5	9236.87949	kg
structural mass of Stage1 of Ariane5	14500.0	kg
structural mass of Stage2 of Ariane5	600.0	kg
total length of Ariane5	98.4	m
Launch Altitude constraint of Ariane5 Constraint	17.0	m
Launch Declination constraint of Ariane5 Constraint	5.2434	°
Launch Longitude constraint of Ariane5 Constraint	-52.7686	°
Launch North Velocity constraint of Ariane5 Constra	0.0	m/s
Launch Radial Velocity constraint of Ariane5 Constr	0.0	m/s
Launch Relative East Velocity constraint of Ariane5	0.0	m/s
Delta of Initial Propellant Mass of Booster_Left_Ta	232728	kg
Delta of Initial Propellant Mass of Booster_Right_T	232728	kg
Delta of Initial Propellant Mass of Stage1_Tank of	145000	kg
Normalized Residual Propellant Mass of Booster_Left	0.0	kg
Normalized Residual Propellant Mass of Booster_Righ	0.0	kg
Normalized Residual Propellant Mass of Stage1_Tank	0.0	kg
Ignition Time of Booster_Left_Prop_Sys of Ariane5 o	0.0	s
Ignition Time of Booster_Right_Prop_Sys of Ariane5	0.0	s
Ignition Time of Stage1_Prop_Sys of Ariane5 of Aria	0.0	s

Real Parameters	value
Yaw constant [rad]	0.6981317
Ignition Time of Booster_Left_Prop_Sys of Ariane5 [s]	0.0
Shut off Time of Booster_Left_Prop_Sys of Ariane5 [s]	123
Ignition Time of Booster_Right_Prop_Sys of Ariane5 [s]	0.0
Shut off Time of Booster_Right_Prop_Sys of Ariane5 [s]	123
Ignition Time of Stage1_Prop_Sys of Ariane5 [s]	0.0
Shut off Time of Stage1_Prop_Sys of Ariane5 [s]	580.0
Payload Scale Factor of Payload_Comp of Ariane5 [-]	1.3195542

Initial Boundary Constraints	value	violation
Launch Altitude [m]	1.0	0.00378591
Launch Longitude [deg]	-52.7686	0.0
Launch Latitude [deg]	5.2434	7.3119E-07
Launch North Velocity [m/s]	0.0	0.0
Launch Relative East Velocity constraint [-] [m/s]	0.0	-3.0615E-16
Launch Velocity constraint [m/s] [m/s]	5.0	0.0
Delta of Initial fuel of tank PROP_TANK_P230_Tank	0.0	0.0
Delta of Initial fuel of tank PROP_TANK_H155_Tank	0.0	0.0

Nonlinear Parameter Constraints	value	violation
P230_Virtual_Ignition_Time [s]	0.0	0.0
H155_Virtual_Ignition_Time [s]	0.0	0.0

```
    Weight of component First_Payload [kg]                    3900.0         0.0

  Final Boundary Constraints                                    value   violation
    Final Lift-Off Altitude constraint [m] [m]                   80.0         0.0

* SCALAR QUANTITIES OF PHASE 2 ************************************************
  Description: Linear Pitch over
  Phase Times (initial + duration = final): 5 + 15 = 20.0

 ...
```

8.6 Result Inspection with External Tools

The results are automatically generated in two formats: `simulation.struct` and `simulation.txt`. These files are placed in the `integ` folder of the scenario. `simulation.struct` is the native output file of the software and it is displayed among others via the *Curve Plots* panel of *Results* tree.

`simulation.txt` is a normal ASCII file that can be loaded in a generic test editor or imported in *Microsoft Excel* and *MATLAB*. Additional export formats are available via the *Exports* node of *Results* tree: e.g. *Google Earth*, *STK*. More details can be found in Section 10.2.

9 Batch Processing

Batch processing is the execution of a series of jobs to complete processes and calculations without manual interaction. Such approach has a number of benefits:

- Less user interaction decreases the idle time of a system.
- Large repeated jobs are done fast in batch systems.
- Large jobs are executed in sequence by the system.
- Many small modifications can be combined in one large job.
- Interaction with other external programs is supported.
- Batch system can be shared by multiple users.

ASTOS provides two different interfaces for batch operations without manual intervention:

1. `Command-line arguments`: predefined command-line arguments allow to run almost all *ASTOS* commands on the command line and allows for automated computations and analysis. The *ASTOS GUI* is not needed to run *ASTOS* on the command line. For more details about supported arguments, please refer to Section 10.4.
2. *Batch-Mode Inspector*: this dedicated *ASTOS GUI* window allows to configure a batch-mode session and store its parameters in a script file. It visualizes the batch structure as a tree whose nodes represent the various process commands/jobs of the batch process. The *Batch-Mode Inspector* can be accessed using the ribbon *Start→Windows→Batch-Mode Inspector* action button. Its use is described in the following chapter.

> **Note:** In the current *ASTOS* version there is no support for a Batch-Mode session to be run on the command line, i.e. a Batch-Mode script file cannot be run on the command line.

9.1 Batch-Mode Inspector: Commands and Functions

The *Batch-Mode Inspector* is a fast and easy way for the user to create and run batch processes in *ASTOS*. It can be accessed via the ribbon *Start→Windows→Batch-Mode Inspector* action button.

Workflow

To evaluate the model for a range of different input conditions, one or more batch variables can be defined which are linked to specific model parameters (see Section 2.10.4.3). The values of these batch variables can then be automatically varied over a user-defined range with the *Batch-Mode Inspector*.
The value of the batch variable at each run is computed according to a user-specified function. The model is then evaluated for each new value of the batch variable which is passed to the linked model parameter. With this approach, the model can be evaluated over a single/multi-dimensional parameter space in a single step.

The value of a batch variable is passed to the model via an intermediate file named `homotopy.xml` file. This file contains the name and the value of each batch variable and is updated by the *Batch-Mode Inspector* at each process run. The `homotopy.xml` file can be viewed directly by the user if required.

Tip: The name of the batch variables defined in the scenario can be copied in memory with *Ctrl+C*; this can be useful when the variable name has to be inserted in the *Batch-Mode Inspector*.

Creating batch sequences

The *Batch-Mode Inspector* provides functionalities to built up a batch sequence from single elements. These sequences can be saved to a file (*File*→ *Save* or *File*→ *Save As...*) to reuse or to share them. The configuration files end with the suffix `.gabc` and can be loaded with *File*→ *Open*. Recently used configurations are listed in the *File* menu as shortcut.
To create a new batch processing press *File*→ *New* in the menu or the **New** button and the root node of the sequence is shown in the tree.

Elements can be added/removed with the help of the *Edit* menu, with the buttons **Add**/**Remove** or directly with a right-click of an element in the tree. When adding an element, a dialog appears that requires the following inputs:

- *Name*: Defines the name of the element to be displayed in the tree.
- *Type*: Selects the desired batch-mode element to be added.
- *Parent*: Selects an existing parent element in the tree.
- *Add after*: Selects the branch of the parent element after which the new element shall be located.

Note: In case of invalid combinations, the **Create** button is greyed out.

The configuration panel on the right can then be used to specify the individual settings of an element as detailed in the following subchapters.

A batch process can be started (stopped) from *Run*→ *Start* (*Run*→ *Stop*) or with the **Start** (**Stop**) button. The batch sequence is generally executed from top to bottom. Specific elements can lead to further branches (*Condition*) or repetitions (e.g. *Loop*) before the next element on the same tree level is taken into account in the process.

Before a batch process starts, an automatic validation process of all elements is carried out. This can also be triggered manually from the menu (*Run*→ *Validate*) or with the **Validate** button. The validation checks the consistency of all specified elements (e.g. if file paths are valid).

To reorder elements in the tree, an element can be dragged and dropped to the desired position with the mouse. *Copy* and *Paste* for selected elements is possible from the right-click menu.

Tip: While a batch process is running, the current progress is indicated with an arrow and some additional information (e.g. a loop counter) is provided in the tree.

9.1.1 Analyse

The *Analyse* batch element allows the user to perform all or a subset of the analyses defined in the scenario. It can be configured as follows:

- If the *Perform all analyses* option is selected, all the analyses are performed.
- Alternatively, in the *Analyses Indices* field the user can specify a discrete set of analyses to be run by entering a list of indices corresponding to the analysis defined in the scenario. The indices must be separated by spaces. For example, if the user wants to run analysis number 1 and 3, he should insert: *1 3*.

9.1.2 Condition

A *Condition* batch element directs a batch process to one of two branches (*true* or *false*) in the process tree depending on a list of conditional expressions. The *true* and *false* branches can be populated with several elements or can be left empty.

One or more conditional expressions can be defined in a table. Each condition is set up in one table row and is evaluated as true or false between two values. The values can be a number or a variable defined in the *Batch-Mode Inspector*. The columns of the setup table have the following meaning:

1. *Composition*: Defines the boolean operator that connects a condition to other conditions (rows). Possible options are *AND* or *OR*.
2. *Var.*: Selects if the first condition value is a variable (if checked) or a number (if unchecked).
3. *Value 1*: Defines the first condition value as explicit number or allows to choose from a list of available variables (dependent on the previous selection).
4. *Rel. Operator*: Defines the relational operator between *Value 1* and *Value 2* Possible operators are: =, <, <=, >, >= or *not equal*.

5. *Var.*: Selects if the second condition value is a variable (if checked) or a number (if unchecked).
6. *Value 2*: Defines the second condition value as explicit number or allows to choose from a list of available variables (dependent on the previous selection).
7. **+**: Adds a new row (i.e. condition) below.
8. **-**: Removes a row (i.e. condition).

If the entire expression is true (i.e. all rows with the provided connections), the batch process follows the *true* branch of the process tree, otherwise it follows the *false* branch.

9.1.3 Counter

The *Counter* batch element defines a special counter variable which increases or decreases with every call. Required inputs are the *Initial value* and the *Increment* of the variable.

Note: The *Increment* is already added to the *Initial value* in the first call.

To refer to the counter value e.g. in other elements, insert *x* there where *x* is the *Counter index* displayed in the element configuration panel.

9.1.4 Evaluate

The *Evaluate* batch element allows the evaluation of a selected parameter in a scenario after the current values of batch mode variables have been applied to the model. The configuration panel provides several settings which specify the source of the parameter:

- *Variable name*: defines an arbitrary unique name for the new variable created from the parameter just evaluated.
- *Source*: selects the source from which the parameter to be evaluated is read. Possible options are *Tops* and *Simulation Struct* files.
- *Use custom STRUCT/TOPS file*: If unchecked, the default scenario files (`input.tops` or `integ\simulation.struct`) serve as data source. When checking this option, also a custom `TOPS` or `STRUCT` file can be used. In this case, the corresponding *Filename* (and path) needs to be specified.

The remaining fields are used to further specify the characteristics of the parameter.

- *Category*: determines the category of the source parameter. Depending on the selected source, the following options are available which also trigger the availability of further specific settings described below:
  - *Independent Variable* (`TOPS` and `STRUCT`)
  - *States* (`TOPS` and `STRUCT`)

- *Controls* (TOPS and STRUCT)
- *Integer Parameters* (TOPS and STRUCT)
- *Real Parameters* (TOPS and STRUCT)
- *Status* (TOPS)
- *Path Constraints* (STRUCT)

> **Note:** For the evaluation of a path constraint the unscaled version of the constraint shall be used. In order to be sure that this is the case, in the *Category* field choose *Auxiliary Functions* and in the *Name* field use the `[Path_Constraint_Name].unscaled` function instead of `[Path_Constraint]`. During optimization the latter is scaled. *Example:* Use `Heat_Flux~Ariane5.unscaled` (*Category*: *Auxiliary Functions*) instead of `Heat_Flux~Ariane5` (*Category*: *Path Constraints*).

- *Initial/Final Boundary Constraints* (STRUCT)
- *Nonlinear Parameter Constraints* (STRUCT)
- *Lagrange/Initial/Terminal Cost* (STRUCT)
- *Auxiliary Functions/Events/Items/Scalars* (STRUCT)

- *Phase*: determines in which phase the parameter value shall be evaluated.
- *Name*: determines the name of the desired parameter. Names must correspond to the name of a quantity as shown for instance in the item table of the *Curve Plots* configuration panel
- *Which Value*: further specifies the value in case the *Category* is not related to a scalar only. Possible options are:
  - *Minimum/Maximum/Average*
  - *Time of Minimum/Maximum*
  - *First/Last*
  - *Upper/Lower Bound*
- *Modifier*: specifies how the parameter value should be treated. Possible options are:
  - *None*: directly evaluated, i.e. no modification
  - *Absolute*: absolute value of parameter value
  - *Negative*: invert the sign of the parameter value
  - *Floor*: round towards minus infinity
  - *Ceiling* round towards plus infinity

9.1.5 Export

The *Export* batch element allows the user to perform all or a subset of the exports defined in the scenario.
It can be configured as follows:

- If the *Perform all exports* option is selected, all the exports are performed.
- Alternatively, in the *Exports Indices* field the user can specify a discrete set of exports to be run by entering a list of indices corresponding to the exports defined in the scenario. The indices must be separated by spaces. For example, if the user wants to run exports number 1 and 3, he should insert: *1 3*.

9.1.6 External

The *External* batch element allows the modification of files with predefined options or to open an external tool, program or script in order to manipulate data. It can be used, for instance, to post-process results using another program or to extract and/or modify data of (intermediary) result files using a script. The *External command* drop-down list already contains templates to copy, move, delete or rename result files. Furthermore, *Custom* commands can be specified.

Tip: For *Linux* and *Mac*, a slash (/) has to be used instead of a backslash (\) in all path definitions.

Copy File

This template requires the *File* to be copied and the location where the copy should be saved (*Copy to path*). The **...** button can be used to browse to the desired file and location.

Note: The target path needs to be already present, the batch process cannot create it automatically.

Move File

This template requires the *File* to be moved and the location where the file should be placed (*Move to path*). The **...** button can be used to browse to the desired file and location.

Note: The target path needs to be already present, the batch process cannot create it automatically.

Delete File

This template only requires the *File* to be deleted. The **...** button can be used to browse to the desired file and location.

Rename File

This template requires the *File* to be renamed and the new filename (*Rename to*).

Note: If a file with the new filename already exists, it is overwritten in the batch process.

Example: Renaming in combination with a *Counter* element
Assume a batch process is used to loop over various simulation runs with a counter element. Each time, an output to a file `.\integ\simulation.txt` is produced. The results for each loop shall however be saved separately. This can be realized with the following setting:
Rename to: *simulation_1.txt*
This renames the default `simulation.txt` produced during the simulation to a filename depending on the value of the counter with index 1, see also Section 9.1.3.

Custom

If *Custom* is selected as *External command*, the file to be executed needs to be provided as *Command* input (together with its path). Optionally, further execution *Parameters* can be specified in the respective field.

9.1.7 Extract

The *Extract* batch element extracts certain data from a simulation file and writes it to another simulation file. First, the specified *Source file* and the *Target file* need to be `STRUCT` file . Then, the *Mode* selects how the extracted data is treated, i.e.:

- *Append* the data to the target file.
- *Replace* the data in the target file.

Category determines what kind of data shall be extracted. Possible options are:

- *Auxiliary Functions*
- *Auxiliary Items*
- *Auxiliary Events*
- *Auxiliary Scalars*

In a final step, the explicit *Name* of the quantity needs to be specified together with the *Phase (Source)*/*Phase (Target)* the data should be extracted from/written to. The name and phase of the quantity must obviously be present in the *Source file*.

9.1.8 Initialize

The *Initialize* batch element initializes the model with new inputs defined by the current batch variable values. The default input and output `TOPS` files are used if the *Use custom TOPS* *file* option is not selected. Alternatively, dedicated *Input TOPS* and *Output TOPS* files can be specified.

Note: This batch element will not work properly when the **Optimization** feature is *disabled*!

9.1.9 Loop

The *Loop* batch element varies the values of a batch variable (see Section 2.10.4.3) in a predefined sequence. First, the corresponding batch variable needs to be linked by specifying its name in the *Variable name* field. The name can also be copied from the *Batch Variables* in the scenario with *Ctrl+C*. Then, different options are available to define the variation range of the values:

- *Start-Steps-Final*: range by initial value, the number of steps and the final value for the variable
- *Start-Step size-Final*: range by the initial value, the step size and the final value for the variable
- *Start-Steps-Step Size*: range by the initial value of the variable, the number of steps and the step size

Note: The number of executed loop iterations always exceed the number of steps by one, as the initial value stage is not counted as a step.

9.1.10 Loop With Values From File

The *Loop with values from file* batch element serves the same purpose as the *Loop* element described in the previous chapter. In this case, however, the batch variable values to be looped over are read from a file.

This *Input file* should be written in plain text (`*.txt`) and numbers need to be arranged column-wise. Multiple columns can be used to represent values for multiple batch variables in one file. Values in one row need to be separated with blank-space or tab key characters.
File data and batch variable(s) are mapped using the provided table:

- *Variable Name*: defines the desired batch variable to be modified.
- *Column Index*: assigns the values of a data column in the file to the batch variable.

It is possible to add or remove assignments to other variables/columns using the **+**/– buttons. Additionally, if an input file contains one or more header lines, an option exists to ignore a defined number of lines at the beginning.

Note: The *Input file* is not accessible during the execution of the loop; no errors are raised, but the file can not be modified (e.g. with a *Save Value*).

9.1.11 Modify Tops

The *Modify TOPS* batch element modifies the value of a model parameter (and its bounds), a constraint or a state. By default, values in the `input.tops` are modified. *Use custom TOPS* however allows specifying any other *Target TOPS* to which the modifications are applied.
A number of additional inputs further specify the model parameter characteristics:

- *Phase*: selects the phase in which the parameter modification shall be applied.
- *Category* select the category to which the parameter belongs. The list includes:
  - *States*
  - *Integer/Real Parameters*
  - *Initial/Final Boundary Constraints*
  - *Path Constraints*
  - *Parameters Constraints*
- *Name*: specifies the name of the model parameter to be modified. Names must correspond to the name of a quantity as shown for instance in the table of the *Curve Plots* configuration panel.

The following settings are further available for states and parameters:

- *Value*: defines the new value of the model parameter.
- *Lower Bound*: sets the new lower bound of the model parameter. Alternatively, the property can be left *Unchanged*.
- *Upper Bound*: sets the new upper bound of the model parameter. Alternatively, the property can be left *Unchanged*.
- *Connected*: sets the connection flag to the previous phase. The parameter can either be *Soft*-, *Hard*- or *Not* connected (see also Section 6.9). Alternatively, the property can be left *Unchanged*.

The following settings are further available for constraints (see also Section 6.9):

- *Equality*: determines if the constraint should be treated as equality constraint (*Yes*), as inequality (*No*) or left *Unchanged*.
- *Enforced*: determines if the constraint should be enforced (*Yes*), not enforced (*No*) or left *Unchanged*.

9.1.12 Optimize

The *Optimize* batch element runs an optimization of the model with the optimization criteria as defined in *Optimization Settings* and the current value of batch variables. The default input and output TOPS files are used. If *Use custom TOPS file* is selected, it is possible to specify different TOPS *Input TOPS* and *Output TOPS* files.

Note: This batch element will not work properly when the **Optimization** feature is *disabled*!

9.1.13 Random

The *Random* batch element can be used to vary the values of a batch variable (see Section 2.10.4.3) in a loop. Different to the *Loop* batch element, no explicit numbers are provided for the variable values, but the values are drawn randomly from a given probability distribution.

To set up the batch element, first the desired *Number of loops* needs to be defined. Then, the distribution parameters can be specified in the provided table:

- *Variable name*: assigns the corresponding batch variable whose values should be varied; the name can also be copied from the *Batch Variables* in the scenario with *Ctrl+C*.
- *Distribution*: specifies the underlying probability distribution of the numbers generated for the variable values. Either *Uniform* or *Gaussian* random numbers can be generated.
- *Lowerbound*: sets the lowest value of the distribution (used for truncation in case of *Gaussian*).
- *Upperbound*: sets the highest value of the distribution (used for truncation in case of *Gaussian*).
- *Mean*: defines the mean value of a *Gaussian* distribution.
- *Standard deviation*: defines the standard deviation a *Gaussian* distribution.

Additional columns (and thus variations for multiple batch variables) can be added/removed with the **+** and – buttons respectively above the table.

9.1.14 Reset Counter

The *Reset Counter* batch element resets the counter value of a *Counter* batch element (Section 9.1.3) to its initial value. Only the *Counter index* of such an element is required as input.

9.1.15 Save Values

The *Save Value* batch element allows saving batch variables in a dedicated file. The variables can be the ones created by *Evaluate* batch elements (Section 9.1.4) or the one used as input variables inside *Loop* and *Random* elements (respectively Section 9.1.9 and Section 9.1.13).

For that purpose, first a *Filename* (and path) needs to be provided to determine where the values should be saved. Then, the *Mode* determines how the data shall be treated:

- *Append*: values are appended after existing data at the end of the target file.
- *New*: a new file is created to store the values.

Output type determines how the data is arranged in the target file. Possible options are *Table* or *Struct*: the first one is a generic ASCII file, the second the native *ASTOS* format.
If *Table* is selected, values are saved in tabular format. The export is further configured using the provided table:

- *Header 1*: defines a header name for the data to be saved (e.g. the variable name specified in the associated *Evaluate* batch element).
- *Variable Name*: selects the source for values to be saved from a drop-down list; the list includes the variables created by *Evaluate* batch elements and the variables used in *Loop* and *Random* elements.

Additional columns (and thus batch variables) can be added or removed with the *Columns* **+**/– buttons above the table. Similarly, also the number of header rows can be modified with the respective *Headers* **+**/– buttons.

If *Struct* is selected, data is saved as a list of values together with their properties in the binary format that can be loaded by *Curve Plots*. In the *Description* above the table, a brief informative description of the data can be provided.
The export is then further configured using the provided table:

- *Name*: determines the parameter name. The name cannot include empty spaces and special character.
- *Description*: adds a brief description of the parameter to be saved, this is used in the axis label when the parameter is plotted.
- *Unit*: specifies the unit associated with the values, please use valid units as defined in Section 2.10.3.2.
- *Variable Name*: selects the source for values to be saved from a drop-down list; the list includes the variables created by *Evaluate* batch elements and the variables used in *Loop* and *Random* elements.

Additional columns (and thus batch variables) can be added or removed with the *Columns:* **+**/– buttons above the table.

9.1.16 Simulate

The *Simulate* batch element runs a simulation of the model with the current batch variable values in a given simulation *Mode*:

- *Multiple Shooting*
- *Single Shooting*
- *Connected Phases*
- *Without Grids*

The *Without Grids* simulation mode is the only mode available when the **Optimization** feature is disabled, e.g. in mission analysis scenarios. This simulation mode is equivalent to running sequentially two actions: an initialization and a simulation with connected phases (see Section 6.5 for details on *Connected Phases*).

Note: Using this batch element with either *Multiple Shooting*, *Single Shooting* or *Connected Phases* will not work properly when the **Optimization** feature is *disabled*!

The default input and result data files are used. If *Use custom TOPS and simulation files* is selected, it is also possible to specify a dedicated *Input TOPS* file and a dedicated *Result Struct-File*/*Result Table-File* to store the simulation results. These result files need to have `.struct` and `.txt` formats respectively to be compatible for later use with the software.

9.1.17 Simulink

The *Simulink* batch element can be used together with the **Simulink_GNC** license feature which interfaces *ASTOS* with *Simulink*. The element starts *MATLAB* automatically (without its GUI) and executes a file (`sim_init.m`) which must be located in the sub-folder `simulink` of the scenario folder (`XXX.gtp`).

The following settings can be specified in the configuration panel:

- *Result Struct-File*: This defines the `STRUCT` file where the results of the simulation with *Simulink* should be saved to.
- *MATLAB path*: This optional setting can be used to set the main path of a certain *MATLAB* installation in case different versions are available on the system.

Please contact service@astos.de for more information about the **Simulink_GNC** license feature and this batch element.

9.1.18 Split Distribution

The *Split Distribution* batch element offers the user the functionality of sorting the elements of an *Auxiliary Item* in ascending order and then saving to a file a certain fraction of the total number of these elements starting with the smallest value.
This functionality can be used for example in the context of a re-entry risk calculation where impact points are generated using a Monte Carlo simulation with normally distributed perturbations on the initial state. Then the *Split Distribution* batch element can be applied in order to obtain an impact probability ellipse for a certain impact probability value. In this case, the most inner impact points (according to a user-defined criterion) which build up a certain percentage of impact probability are saved in the file specified in the settings of this batch element.

- In the *Filename* setting the user shall provide the path of and the file name which contains the input data. This file should have been created using the *Save Values* batch element before using the *Split Distribution* batch element. The file will be overwritten with the data resulting from the calculations of the *Split Distribution* batch element. The user can use the *External* batch item in order to save the input data to a new file in case there is need for this.
- In the *Criterion* setting the user shall provide the name of the *Auxiliary Item* (see Section 8.1 for details on *Auxiliary Item*) which accommodates the data to be sorted and split by the *Split Distribution* item.
- Using the *Fraction* setting the user can specify a fraction between 0.0 und 1.0. After the *Auxiliary Item* values have been sorted in ascending order, as many values will be saved to the *Filename* file starting with the smallest value, as indicated by the *Fraction* of the total number of the *Auxiliary Item* values. For example, for the following already sorted list of *Auxiliary Item* values {11.2, 12.5, 29.3, 123.9, 144.0, 234.6, 354.76, 787.9856} and for a *Fraction* value of 0.23, the algorithm computes that the first *ceil(8*0.23) = ceil(1.84) = 2* data values starting with the smallest value of the interval will be saved to the file, i.e. {11.2, 12.5}.

9.2 Batch-Mode Inspector: Example

The scenario contains a simulation of the *MASER 11* sounding rocket. The *MASER* program is conducted by Swedish Space Corporation (SSC) from Kiruna. It uses a two-stage solid propellant rocket. Earlier, *Black Brant IX* in different versions or the *Skylark 7* have been used. They are now replaced by the *VSB 30* rocket motor. A guidance system, *S-19*, is used depending on the mission requirements. The scenario is intended only for simulation purpose with a batch procedure to automate the ASTOS action schedule.

Sounding Rocket Model

Several complex models are presented in this scenario, all of them are accessible from the Modelling tab in the ASTOS 8 main window.

- 6DoF aerodynamics: coefficients are provided to include axial, normal and later forces as well as moments about the three main axes of the vehicle. This includes also fin deflection.
- 6-DoF equation of motion: the position and the attitude of the vehicle (together with their derivatives) are integrated along the trajectory.
- Thrusters to induce a roll moment after separation from the rail.
- Special EoM for launch-pad and rail movement, respectively in phase 1 and 2.
- Maneuver at the beginning of the third phase and de-spin at the beginning of the experimental phase (yo-yo effect).
- Parachute model as special aerodynamics with variable reference area.

Preparation

- Open ASTOS.
- From the *Scenario* menu, select *New...*.
- Insert in the *New Scenario* cell the path where you want to place the working case (e.g. "`C:\work\TC_MASER11.gtp` ").
- Activate the *Copy from existing problem* option and insert the path of the scenario: `%ASTOS%\examples\Analysis\Mission Performance\MASER11.gtp` (where `%ASTOS%` is the installation path of ASTOS).
- Select *Create*.

- In the *Modelling* tree of the *ASTOS* GUI inspect the scenario characteristics, in particular the *Aerodynamics* and *Phase Settings*.
- Open the Batch-Mode Inspector, using the *Start*→*Windows*→*Batch-Mode Inspector* action button.
- From the Batch-Mode Inspector, open the `batch.gabc` file with the *File*→ *Open...* command.
- Expand the "Random_1" element, to visualize the actions performed by the batch process.
- Select the **Start** button.

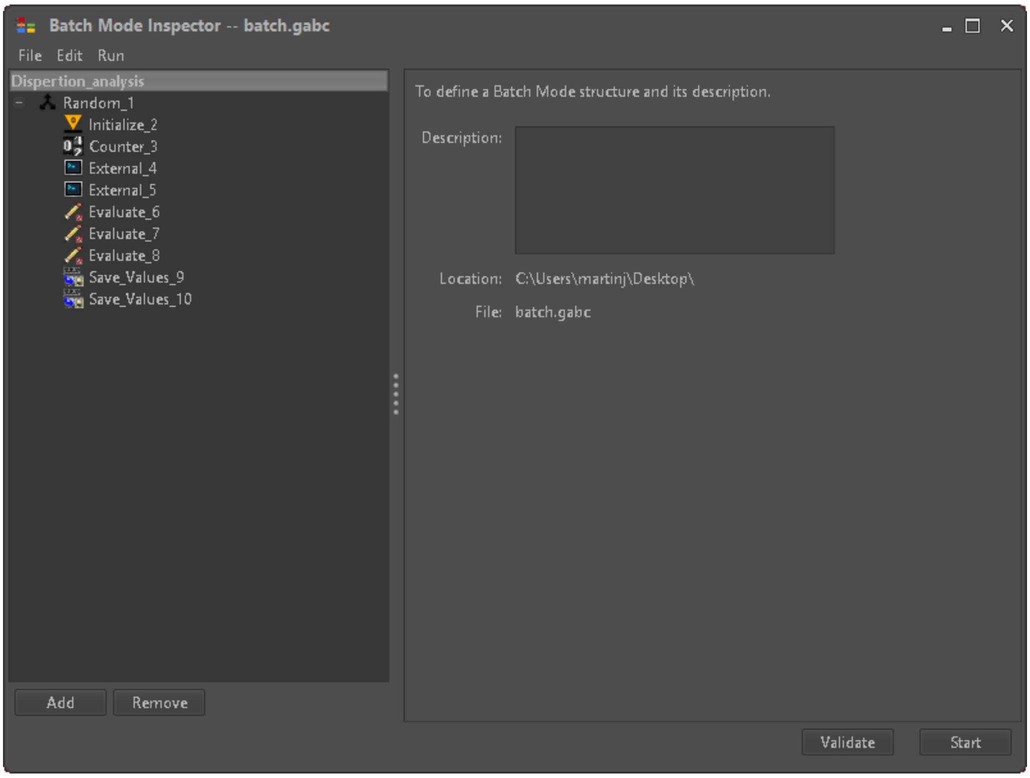

Fig. 9.1: Batch mode scenario overview

During the post-process, the default `simulation.txt` file is copied to a different folder and renamed with the step number (via the counter). Then the final latitude, the final longitude and the maximum altitude at the apogee are extracted and saved in the `summary.struct`. Finally all values of the random input batch variables used to produce the result are stored in a second file `summary.txt`.

Note: the simulation_x.txt files are saved in the "batch" sub-folder, the process will overwrite files already present with the same name.

Explanation of the different steps

- *Random_1*
  Ten loops will be performed with different random input batch variables. In each step new random values for the scaling factors of the aerodynamic coefficients, Isp of the two stages, the payload mass etc. are considered. Two different kinds of random distributions can be selected:

  - *Gaussian*
  - *Uniform*

  The *Uniform* distribution will generate random values with uniform distribution within the lower and upper bound. For the *Gaussian* distribution you have to select a mean value which is typically the nominal value of the variable (e.g. 1.0 for the scaling factors). Then a standard deviation from this mean value has to be selected. 68.2% of all random values will be in the area of the mean value plus/minus the standard deviation. The lower and upper bound are typically selected to be the mean value plus/minus three times the standard deviation. This ensures not to get randomly values far off the mean value.

- *Initialize_2*
  After the batch variables have been randomly selected ASTOS initializes with these variables and simulates the trajectory.
- *Counter_3*
  Counts the iterations and stores the actual iteration number in a variable that will be used later to rename the simulation files.
- *External_4*
  Copies the `simulation.txt` from the `.\integ\` into the `.\batch\` folder.
- *External_5*
  Renames the copied `simulation.txt` in the `.\batch\` folder into `simulation_$current_loop_value$.txt`.
- *Evaluate_6-8*
  The next three evaluation steps evaluate the values for the final longitude, final latitude as well as the max. apogee altitude and save them into a variable.
- *Save_Values_9*
  Creates a `summary.txt` file in which the random values of the batch variables for each iteration are stored.
- *Save_Values_10*
  Creates a `summary.struct` file in which the evaluated values of final longitude, final latitude and max. apogee altitude for each loop are stored. This file can be loaded into the Viewer of ASTOS later on to compare the results of all iterations.

10 Integration of ASTOS in a Working Environment

The design of a modern space mission is performed in the Concurrent Design Facility (CDF) where each mission sub-system is managed by a responsible person using a computer running a set of software specific for this particular sub-system. *ASTOS* is being used at such facilities and it is therefore capable of importing data in various formats (e.g. output from other software) and exporting computation results to multiple formats which can be input for other software. Moreover, the actions performed by *ASTOS* (e.g. initializion, optimization or analyses) can be run from the command line without the need of the GUI such that *ASTOS* can be integrated in an automated environment. The GUI is required for the comfortable modelling of a scenario by the user and is very often used to also perform the software actions mentioned above.

10.1 Configuration of Import Data

The scenario definition includes inputs coming from several subsystems. Usually, these inputs are not provided by a single person, but by a team where one person is responsible only for one certain subsystem. *ASTOS* is well suited to be the central place to collect all the required inputs. Amongst others, it can read data stored in generic text files, in *MySQL* databases or in specific formats as e.g. *Excel*, `TLE`, `obj` or `spice`.

10.1.1 Database Variables

The definition of variables is a useful tool to simplify the management of an extensive amount of data by supporting external software: instead of linking an input cell or table to an internal variable, it is also possible to link them to a *Microsoft Excel* cell or to a *MySQL* database. The definition of variables is detailed in Section 2.10.4.2.

10.1.2 Profile Data Import

The profile data import manages the reading of several ASCII external files and provides the following features:

- Data definition by columns
- Special file formats for 2- and 3- dimensional data
- Variable definition of separators: tabulator, comma, semicolon, space
- Optional identifier row
- Optional description row
- Support of spreadsheet application conformed format
- Format determination by the file name extension

Description

Column-defined data can be auto-detected by the extension of the file name. The number format always uses US English settings with a dot as decimal separator. Most kinds of number formats can be read and various number formats can be present in a file. Files can be read in *DOS* and *Unix* format. The maximum number of columns is limited to 256. Tabulator, space, comma and semicolon are supported as column separator, however the same separator needs to be used throughout the file. A file may contain additional information in the first lines which need to be skipped by defining *Skip_Lines* properly. Please do not skip any header line required by the data format itself, such as column identifiers or the values of the independent variables. More information is found in the corresponding sub-sections about the data formats.

Note: Matrix-defined data must be rectangular, and the selector type must be set to *Identifier* (setting it to *Column* raises an error).

Diagnostics

The following errors are printed in the *Execution log* window in case the data file cannot be processed and loaded.

Table 10.1: Import format errors and their meaning

`Data_Error` or `Format_Error`	The file contains characters which cannot be processed or the number format is not readable
`End_Error`	Unexpected end of file
`Illegal_Identifier_Error`	An identifier is used, which is not present in the file

10.1.2.1 Import of 1-Dimensional Data

One-dimensional data refers to data that depends on one independent variable. It can be defined in the character-separated values (.csv) format which is column-based and which is a common export format of spreadsheet applications.

Description of the character-separated format

Data files of this class can be auto-detected, if they have one of the extensions: `.txt`, `.dat` or `.csv` (CSV is a special file format with comma separated values used by *Microsoft* applications). The columns of this file type can be separated by different characters: semicolon, tabulator, space. In some special cases also a comma is possible. If a space is not the column separator, leading spaces are ignored.

The first column of a 1-dimensional data file contains the values of the independent variable. Each of the following columns contains the values of a certain data variable, one value per line and corresponding to one value from the column of the independent variable. Thus each row contains as a first data item one value of the independent variable and as many other data items as the number of data variables. As a consequence the data in the 1-dimensional data file represents a rectangular matrix. The file can contain up to three header lines for identifier, description and unit of each column. If header rows exist, then the first one must contain as many identifier strings as columns exist. A column identifier cannot contain spaces, instead it is suggested to use underscores if applicable (e.g. `Mach_Number`). An optional second header line contains the descriptions to the identifiers. If the column separator is a space, then the description string must be quoted, e.g. `"My description"`. An optional third header row contains the units of the columns. In case a column does not have a unit, the unit name `None` shall be used. Column identifiers are used to select the columns to be used as independent variable and data value. Instead of an identifier the column index (integer number) can also be used. Each file may contain more data columns than specified in the profile data table.

Empty rows are not allowed. The end of file (EOF) character can be placed after the last number or at the beginning of the next line. Each line can be finalized by any combinations of carriage return (CR)/ line feed (LF) after the last number in the row. Refer also to Section 10.1.2 for general format rules.

Example

A valid ASCII file containing 1-Dimensional data is shown in Fig. 10.1. Here the first row contains the column identifiers. The data columns delimiter (i.e. tab) as well as the line end characters are also shown.

```
Mach	CY
0.1	-0.3714
0.3	-0.3774
0.5	-0.3833
0.9	-0.4062
1.1	-0.4875
1.35	-0.4385
1.5	-0.3726
2.0	-0.3159
2.7	-0.2561
3.0	-0.2368
3.5	-0.2121
4.0	-0.1939
4.5	-0.1799
5.0	-0.1799
5.5	-0.1799
6.0	-0.1799
6.5	-0.1799
```

Fig. 10.1: 1-Dimensional data file.

10.1.2.2 Import of 2-Dimensional Data

Two-dimensional data can be defined in a column-based format or a special matrix format. Data defined in the matrix format must be rectangular. Column-based formats are described in detail in Section 10.1.2.1 and the extensions for non-rectangular data in Section 10.1.2.4.

Description of the matrix format

The first row contains *n* columns of the *x*-dimension vector and the second row *m* columns of the *y*-dimension vector. The following rows represent *a m x n* matrix with *m* rows and *n* columns. The complete file must consequently contain *2+m* data rows. The data file must not contain any strings (e.g. identifiers) and empty rows are not allowed as well. The end of file (EOF) character can be placed after the last number or in the beginning of the next line. Each line can be finalized by any combinations of CR/LF after the last number in the row. Refer also to Section 10.1.2 for general format conditions.

Example 1: 4 x 5 data matrix

The tabular data

x / y	0	0.50	1.00	1.30	1.50
1.00	0.11	0.12	0.13	0.14	0.15
2.00	0.21	0.22	0.23	0.24	0.25
3.00	0.31	0.32	0.33	0.34	0.35
4.00	0.41	0.42	0.43	0.44	0.45

Fig. 10.2: A 4x5 matrix.

should be formatted in a file as follows:

```
0.00 0.50 1.00 1.30 1.50      - 1st indep. variable x
1.00 2.00 3.00 4.00           - 2nd indep. variable y
0.11 0.12 0.13 0.14 0.15     \
0.21 0.22 0.23 0.24 0.25      | x-y matrix
0.31 0.32 0.33 0.34 0.35      |
0.41 0.42 0.43 0.44 0.45     /
```

Fig. 10.3: A 4x5 matrix format in a file ready to be imported

Example 2: Drag coefficient data

In this example it is shown how to import a 2-dimensional table describing the CD as a function of the altitude and the Mach number. This can only be realized using a `.txt` file. Assume the data is stored in a file `CD_S1_thrust_boosters.txt` in the `model` sub-folder of the scenario. The structure of the file is as described above, with each column having a different altitude (1st indep. variable) and each row having a different Mach number (2nd indep. variable).

1. Navigate to the *Profile* section in the configuration panel of an aerodynamics models in the *Modelling* tree (make sure that the *Cd* is *Defined by Profile*).
2. Choose *File* as *Data source*.
3. Insert the name of the file in the *Filename* cell (*.\model\CD_S1_thrust_boosters.txt*).
4. In the interpolation data table insert an additional column left to the *Data* column with the appropriate button.
5. The first independent variable has to be mapped to the first column, the second independent variable to the second column. Thus, select the *Name* of the first column cell and select *Altitude* from the drop-down menu. The respective unit *Unit* cell is automatically updated to *Kilo-meter*.
6. Repeat the same procedure for the second column selecting *Mach* and as independent variable and *None* as unit.
7. The third row of the table is called *Column*. Click on this row header and it changes to *Identifier*. This is the correct format for multidimensional tables, since *Column* is only applicable to column format data.

10.1.2.3 Import of 3-Dimensional Data

Three-dimensional data can be defined in a column-based format or a special matrix format. Data defined in the matrix format must be rectangular. Column-based formats are described in detail in Section 10.1.2.1 and the extensions for non-rectangular data in Section 10.1.2.4.

Description of the matrix format

The first row contains *n* columns of the *x*-dimension vector. The second row contains *m* columns of the *y*-dimension vector and the third one *k* columns of the *z*-dimension vector. Then for each element of the *z*-vector, a *m x n* matrix must follow. Each row must contain *n* columns. Consequently, the complete file must contain *3+m*k* data rows. Empty rows at the end are ignored, but not allowed between the data rows. The data file must not contain any strings (e.g. identifiers).

Note: As for the 2D matrix format, the third row in the profile data table object table has to be set to *Identifier*, not *Column*.

Empty rows are not allowed. The end of file (EOF) character can stay after the last number or in the beginning of the next line. Each line can be finalized by any combinations of CR/LF after the last number in the row.

Example: 4 x 5 x 2 matrix data

The tabular data

Z = 10.00

y \ x	0.000	0.500	1.000	1.300	1.500
1.000	0.111	0.112	0.113	0.114	0.115
2.000	0.121	0.122	0.123	0.124	0.125
3.000	0.131	0.132	0.133	0.134	0.135
4.000	0.141	0.142	0.143	0.144	0.145

Z = 20.00

y \ x	0.000	0.500	1.000	1.300	1.500
1.000	0.211	0.212	0.213	0.214	0.215
2.000	0.221	0.222	0.223	0.224	0.225
3.000	0.231	0.232	0.233	0.234	0.235
4.000	0.241	0.242	0.243	0.244	0.245

Fig. 10.4: A 4 x 5 x 2 matrix

should be formatted in a file as follows:

```
0.000 0.500 1.000 1.300 1.500      - 1st indep. variable x
1.000 2.000 3.000 4.000            - 2nd indep. variable y
10.00 20.00                        - 3rd  indep. variable z
0.111 0.112 0.113 0.114 0.115     \
0.121 0.122 0.123 0.124 0.125      | x-y matrix
0.131 0.132 0.133 0.134 0.135      | for z=10.00
0.141 0.142 0.143 0.144 0.145     /
0.211 0.212 0.213 0.214 0.215     \
0.221 0.222 0.223 0.224 0.225      | x-y matrix
0.231 0.232 0.233 0.234 0.235      | for z=20.00
0.241 0.242 0.243 0.244 0.245     /
```

Fig. 10.5: A 4 x 5 x 2 matrix format in a file ready to be imported

10.1.2.4 Import of Non-Rectangular Data

Non-rectangular data can be used together with *General B Spline* as *Interpolation type*. Empty rows are not allowed in the import file. The end of file (EOF) character can be placed after the last number or in the beginning of the next line. Each line can be finalized by any combinations of CR/LF after the last number in the row.

Example: 2-dimensional non-rectangular data

The following file content serves as an example for non-rectangular data, i.e. data values (3rd column) are not specified for each value pair of the independent variables (1st and 2nd column).

```
1.0, 0.1, 0.11
1.0, 0.2, 0.12
2.0, 0.1, 0.21
2.0, 0.2, 0.22
2.0, 0.3, 0.23
3.0, 0.2, 0.32
3.0, 0.3, 0.33
```

Above would result in the following band structure:

Table 10.2: Non-rectangular band structure

	0.1	**0.2**	**0.3**
1.0	0.11	0.12	
2.0	0.12	0.22	0.23
3.0		0.32	0.33

10.1.3 Model-Specific Data Import Formats

The are several import formats that are supported by *ASTOS*. Most of them are required to handle tabular data for the definition of profiles. Other specific formats are described in details together with the associated model. Examples for such formats are:

- `C S Table` is a set of cosine and sine coefficients used by the *Spherical Harmonics* gravity definition (refer to chapter Celestial Body in Model Reference).
- *SPICE* ephemeris files (`*.bsp`) can be used to define the ephemerides of celestial bodies (see chapter Celestial Body in Model Reference).
- `ODIN_MER` mass estimation tables generated by *ODIN* can be used for the structural mass approximation of detail design components (see chapter Mass Computation in Model Reference).
- `TLE` format can be used in the Navigation Analysis and as Initial State (see Model Reference).
- *Wavefront OBJ* (`*.obj`) and *3DS Max* (`*.3ds`) are supported mesh file formats that can be used in the *Vehicle Structure* (refer to Section 4.4.1). The settings for exporting an *Wavefront OBJ* file with *Blender* are provided below in this section.
- *RPA* configuration files (`*.cfg`) can be used for the setup and analysis of a *Throat Design* engine (see chapter Throat Design Rocket in Model Reference).
- *SPECS* files (`*.spc`) allow to specify additional parameters for optimizers and NLP solvers (see Section 6.6.6).

CCSDS Ephemeris

Two *CCSDS*-recommended ephemeris messages are described in this chapter: the Attitude Ephemeris Message (*AEM*) and the Orbit Ephemeris Message (*OEM*). Ephemeris data in CCSDS format can be used to describe the vehicle dynamics (*Modelling* tree: *Dynamics Configuration -> Vehicles & POIs Dynamics*); in particular OEM is used as *Equations of*

Motion (if *State from File* is selected) and AEM as *Attitude* (if *State from File* is selected). The recommended data messages are ASCII text format (`*.aem` for AEM and `*.oem` for OEM). While binary-based data message formats are computer efficient and minimize overhead, there are applications for which an ASCII character-based message is more appropriate. For example, when files or data objects are created using text editors, ASCII character-based data format representations are necessary. Besides, they are also useful in transferring text files between computer systems since the ASCII character set is nearly universally used. In addition, direct human-readable downloads of text files or objects to displays or printers are possible without preprocessing. The penalty for this convenience is inefficiency.

Attitude Ephemeris Message

An *AEM* specifies the attitude state of a single object at multiple epochs, contained within a specified time range. The AEM is suited to exchanges that

1. involve automated interaction (e.g., computer-to-computer communication where frequent, fast, automated time interpretation and processing are required), and
2. require high fidelity or high precision dynamic modeling (e.g., flexible structures, more complex attitude movement, etc.).

The AEM allows for dynamic modeling of any number of torques (solar pressure, atmospheric torques, magnetics, etc.). Attitude Ephemeris Message requires the use of an interpolation technique to interpret the attitude state at times different from the tabular epochs. It is fully self-contained and no additional information is required when inertial reference frames are specified. If local orbital reference frames are specified, then an AEM must be used in conjunction with an *OEM* (see below).

More information about CCSDS AEM can be found in the recommended standard [37].

Orbit Ephemeris Message

An *OEM* specifies the position and velocity of a single object at multiple epochs contained within a specified time range. The OEM is suited to exchanges that

1. involve automated interaction (e.g., computer-to-computer communication where frequent, fast automated time interpretation and processing is required), and
2. require high fidelity or high precision dynamic modeling.

The OEM allows for dynamic modeling of any number of gravitational and non-gravitational accelerations. It requires the use of an interpolation technique to interpret the position and velocity at times different from the tabular epochs. Orbit Ephemeris Message also contains an optional covariance matrix which reflects the uncertainty of the orbit solution used to generate states in the ephemeris.

More information about CCSDS OEM can be found in the recommended standard [38].

Wavefront OBJ

The user-defined 3D mesh can be created with an external tool (e.g. *Blender*) or with the internal *CAD Import and Texturing Tool*. For a detailed description of this tool, see Section 4.4.2). If instead an external tool is preferred, please consider that only triangular faces are supported. Hereafter a list of the settings for *Export OBJ* in *Blender* v.2.78 are provided:

- Forward: +Y Forward
- Up: Z up
- Apply Modifiers
- Write Normals
- Include UVs
- Write Materials
- Triangulate Faces
- Objects as OBJ Groups
- Scale 1.00, since *Astroview* is using meter for the visualization.

10.2 Results Exports

This item allows the user to export simulation data to a file that can be used by another program. The *Simulation input file* to be exported can be selected by the user. To create an export template it is possible to use the ribbon *Add*→ *Post-Processing*→ *Export*. The *Add Export* appears to select the *Identifier* (i.e. the file name) and the *Type* (i.e. data formats) as well as the *Subtype*. The available data formats are:

- *CCSDS Navigation Message / Parameter Message*, a file (`*.opm`) is generated reporting the state at the requested phase and evaluation time (i.e. epoch) in line with the CCSDS standards [38]. If requested, a file (`*.apm`) is generated storing the rotational state (*Attitude Data Message*) at the same epoch. The format of this file is described in [37]. Further information about the CCSDS data format can be found in Section 10.1.3.
- *CCSDS Navigation Message / Ephemeris Message*, a file (`*.oem`) is generated reporting the states of the selected vehicle along the whole trajectory in line with the CCSDS standards [38]. If requested, a file (`*.aem`) is generated storing the rotational states (*Attitude Data Message*) at the same epochs. The format of this file is described in [37]. Further information about the CCSDS data format can be found in Section 10.1.3.
- *Excel*, it places single values or arrays of data in user specified cells within an existing `*.xls` or `*.xlsx` file.

  > **Tip:** Before performing the create export action of an *Excel* export (either from the GUI or the command line) please check that the excel file already exists.

- *GESOP Structure* exports the *ASTOS* output data file in various formats depending on the user setting. This can be one of the following:
  - *Struct* for the default *ASTOS* output data file `.struct`
  - *XML* for *ASTOS* output data file in human readable format `.xml`
  - *MATLAB m-file* generates a *MATLAB* `.m` file containing a function with the same name as the name provided for the `.m` file in the *Output filename* setting. The `.m` file ending is

added to the file name automatically and the file name shall not contain any spaces or special characters. When executed in the *MATLAB* environment, the function returns the content of the *ASTOS* output data, both values (in SI units) and structural information.

- *Google Earth*, the created file contains the position data plus some additional data, i.e. phase number, impact points etc. (`*.kml`)
- *MySQL*, it places single values of data in user specified data table in line with one or more primary keys.
- *STK animation*, three files are created: the ephemeris, the attitude and the articulation file (`*.e`, `*.a`, `*.lvma`); these are compatible with STK version 10.0.
- *Text*, tabulator separated format (`*.txt`)
- *MBS*, the created files contain the multibody models of the scenario at specific points in time. Further information is given in chapter Linearised Flexible Dynamics in the Multibody and Flexible Dynamics book

In case of STK the articulation file needs to be manually adapted to the *STK* 3D model (e.g. `Ariane5.mdl`); since the articulation names defined by *ASTOS* (according to the stage and propulsion names defined in the vehicle) could be different from the articulation names defined in the *STK* model.

Note: This export has been prepared for the actions defined in the Ariane 5 model of *STK* (i.e. length, size, separate, drop and rotate), different *STK* models could present different actions. Please contact service@astos.de in case a support is needed.

10.3 Databases and ODBC Connections

MySQL is one of the most popular open source database management system. It is an important tool for an efficient workflow with large amounts of data to process and access. For that reason *ASTOS* is able to access also *MySQL* databases. Numeric input fields or tables can be defined as a variable (see Section 2.10.4.2) whose values can be read from a database table. A second option to account for external values is to read data from *Microsoft Excel*. Numeric input field or tables which are defined as a variable (see Section 2.10.4.2) can access values from *Excel* sheets.

10.3.1 MySQL Database

First, *ASTOS* needs access to the database. The configuration dialog can be started using the ribbon *Start→Windows→Database Connections* action button. To set up a connection to a *MySQL* database, information about both *Database credentials* and *Connections* has to be specified in the *Database Connections* dialog.

Database credentials

Login *Database credentials* can be added and removed by using the **Add**/**Remove** buttons. For each credential, a combination of *User name* and *Password* has to be specified in the respective table cell. The *Credential ID* is an arbitrary but unique identifier used to assign the credential to connection settings in a later step.

As an additional security option, it is possible to create a *Key file* which is used to encrypt passwords of database users. The key file can be stored anywhere (e.g. in the `.gtp` folder) and prevents unauthorized access to a database (i.e. if the key file is not present or in a secure location).

To create a key file, click the **Create Key File** button and save it at the desired location. To reuse an existing key file, either its file name and path can manually be entered in the input field or the **...** button can be used to navigate to the file location.

Note: If a user shares a complete scenario folder (`.gtp`) that contains credentials, also the key file has to be provided. Otherwise passwords cannot be decrypted by *ASTOS* and the database connections fails.

Connections

The *Connections* panel is used to specify the individual databases. To add a database, press the **Add** button, select *Database connection* in the *New Connection* dialog and press **Create**. **Remove** deletes an existing entry from the table.

As it is possible to set up multiple connections, each entry has an arbitrary but unique *ID*. These identifiers are used later on in the linking procedure for database variables (see Section 2.10.4.2). The *DB Connector* is automatically determined (*MySQL Connector* in this case) and cannot be modified. User input is required in the following columns to define the database to be accessed:

- *Location* defines the database location on a server (also via IP address)
- *Port* defines the database server port
- *DB Name* defines the database name at the specified location
- *Credential ID* provides access to the database

The *Credential ID* can be selected from a drop-down list in the respective cell that contains all previously defined credentials.

Note: Before creating a *MySQL* database connection, credentials with username and password for the database have to be defined.

The button **Test Selected** (or **Test All**) can be used to determine if a selected (or all) databases can successfully be accessed.

Once a database connection has been established and data has been linked to variables of a scenario, the respective values are read from the database. By default, these values are not automatically updated when the database is externally modified. In such cases, use the **Update Selected** or **Update All** buttons to update all variable values linked to a selected (or all) database(s).

Note: Use the buttons **Update Selected** or **Update All** after databases have externally been modified to account for the changes in *ASTOS*.

10.3.2 Microsoft Excel File

The configuration dialog in order to access an *Excel* sheet can be started using the ribbon *Start→Windows→Database Connections* action button. The *Database credentials* panel in the *Database Connections* dialog is only relevant for *MySQL* connections and thus requires no inputs in this case.

To add a connection to a spreadsheet, press the **Add** button in the *Connections* panel, select *Excel file connection* in the *New Connection* dialog and press **Create**. **Remove** deletes an existing entry from the table.

As it is possible to set up multiple connections, each entry has an arbitrary but unique *ID*. These identifiers are used later on in the linking procedure for database variables (see Section 2.10.4.2). The *DB Connector* is automatically determined (*EXCEL File Connector* in this case) and cannot be modified.

Further user input is required only in the *Location* column. Here, the path and file name can either be directly entered or the **...** button that appears in the selected cell can be used to navigate to the file location.

The button **Test Selected** (or **Test All**) can be used to determine if a selected (or all) files can successfully be accessed.

Once a file connection has been established and data has been linked to variables of a scenario, the respective values are read from the spreadsheet. By default, these values are not automatically updated when the file is externally modified. In such cases, use the **Update Selected** (or **Update All**) buttons to update all variable values linked to a selected (or all) file(s).

Tip: Use the buttons **Update Selected** or **Update All** after a spreadsheet has externally been modified to account for the changes in *ASTOS*.

10.4 Command Line Arguments

In addition to the operation of the *ASTOS* software via the GUI, it can be also operated from the command line. Once a scenario has been configured and saved, the software behaviour and the actions to be performed can also be controlled from the command line. In order to open a command line window (prompt / terminal), on *Microsoft Windows* the `cmd.exe` can be run and on *Linux/Mac* the shortcut Ctrl+Alt+T can be used. There are two two command line interfaces available:

- *ASTOS GUI* interface
- *ASTOS/GESOP* Core interface

The supported command line arguments by both interfaces are explained in the following sections. In the examples below `%ASTOS%` represents the installation folder of the *ASTOS* software and `%Scenario_Path%` stands for the full path of the scenario .gtp folder.

ASTOS Start Script Parameters

The command to start an operation using the *ASTOS GUI* interface on the command line is

- *Microsoft Windows*: *%ASTOS%*`\start.bat [-argument]`
- *Linux/Mac*: *%ASTOS%*`/start.sh [-argument]`

Running *start.bat* starts the GUI and does not perform any actions. The following table gives a list of all corresponding command line arguments and their brief explanation.

Table 10.3: Command line arguments of the GUI interface

Argument	Description
`-?`	show all available command line arguments
`-analyses`	create all analyses defined in a specified scenario
`-bm:<file_name>.gabc`	run the software in Batch-Mode using the file <file_name>.gabc.
`-execlog:<file_name>`	write execution output into file <file_name>
`-exports`	create all exports defined in a specified scenario. The index count starts at 1. A list of the supported export types when *ASTOS* is used on the command line is given in Table 10.7 .
`-exports:{export id, ...}`	create all exports in a specified scenario with defined report id. The index count starts at 1. A list of the supported export types when

Argument	Description
	ASTOS is used on the command line is given in Table 10.7 .
-exports:[{index, ...}]	create all exports in a scenario with the specified report index. The index count starts at 1. A list of the supported export types when *ASTOS* is used on the command line is given in Table 10.7 .
-initialize	run an initialization task
-optimize	run an optimization task (cold start)
-optimize:cold	run optimization: cold start
-optimize:warm	run optimization: warm start
-pause	causes the >GUI to exit after pressing the [Enter] key
-reports	causes that all reports defined in a specified scenario are created
-reports:{report id, ...}	create all reports with defined report id
-reports:[{index, ...}]	create all reports with defined report index; index starts from 1
-result_summary	create a result summary
-scenario:<scenario_folder_path>	load and use the scenario <scenario_folder_path>
-simulate	run a simulation task (by default with multiple shooting)
-simulate:single	run a single shooting simulation task
-simulate:multi	run a multiple shooting simulation task
-simulate:connected	run a connected phase simulation task
-tops:<file_name>.tops	load a TOPS file named <file_name>.tops
-update_all_databases	updates all defined database and excel variable values and saves the scenario
-update_model_data	update model data
-verification	run verification process
-write_scenario	enforce writing the scenario at the end

Note: In order to run *ASTOS* on the command line at least two arguments have to provided. One of these argument must be -scenario:<scenario_folder_path>.

Note: If an argument value contains blank spaces then the whole argument has to be surrounded by double quotes ("), e.g. %ASTOS%\start.bat "-scenario:C:\My Folder \myscenario.gtp" -exports.

Tip: All arguments which require a filename (e.g. -tops) accept relative paths with respect to scenario folder. A relative path starting with a point "." specifies a location with respect to the scenario directory, while starting with two points ".." specifies a location with respect to the scenario parent directory.

The execution output will be saved in file exec_log.txt but it can be also redirect using the -execlog:<file_name> argument.

ASTOS/GESOP Parameters

The tasks which can be run from the *ASTOS GUI* (e.g. *Initialization*, *Simulation* or *Optimization*) can be started on the command line using the *ASTOS/GESOP* Core executable

- *Microsoft Windows* : %ASTOS%\bin\gesop.exe [-argument]
- *Linux/Mac*: %ASTOS%/bin/gesop [-argument]

The following table gives a list of all corresponding command line arguments which cause that a task (action) is runn. A brief explanation is provided for each argument.

Table 10.4: Command line arguments of the ASTOS/GESOP Core Interface which trigger a task (action)

<table>
<tr><th>Argument</th><th>Description</th></tr>
<tr><td>-I</td><td>runs a Simulate task using given control laws (with inactive Optimization feature only)</td></tr>
<tr><td>-G (d|r|i|n|v)</td><td>generate a description, read trajectory, use initial guess, nullify grid or verify an existing TOPS file (with active Optimization feature only)</td></tr>
<tr><td>-O (c|w)</td><td>starts an Optimization task using Cold Start or Warm Start (with active Optimization feature only)</td></tr>
<tr><td>-S (m|s|c)</td><td>starts a Simulation task using multiple / single shooting or connected phases (with active Optimization feature only)</td></tr>
<tr><td>-P</td><td>creates a scenario summary *.html file in the scenario .gtp folder</td></tr>
<tr><td>-U</td><td>updates old scenario structure</td></tr>
<tr><td>-E (a|A)|((1|2|3|4|5|6|7|8|9)","(0|1|2|3|4|5|6|7|8|9)*)</td><td>starts an Export task using all exports or comma separated list of indices. A list of the supported export types when ASTOS is used on the command line is given in Table 10.7 .</td></tr>
<tr><td>-A (a|A)|((1|2|3|4|5|6|7|8|9)","(0|1|2|3|4|5|6|7|8|9)*)</td><td>starts an Analyse task using all analyses or comma separated list of indices</td></tr>
</table>

There is a set of common *ASTOS/GESOP* Core command line parameters which do not result in running a task. The following table contains these parameters and their brief explanation.

Table 10.5: Common command line options and their short description

Argument	Description
-(h|H)	shows the command line help. If an Action argument from the table above precedes this argument, a list of compatible and required options corresponding to this Action argument is displayed
-c	sets the working directory. The working directory of the *ASTOS* software is the scenario .gtp folder.
-DCAP	sets the installation directory of the *DCAP* software. It is only required for multibody dynamics.
-m (Xml|Binary_Token|Ascii_Token|Data_Stream)	sets the output mode
-i <TOPS_file_location>	specifies the input TOPS file and the path to it
-o <TOPS_file_location>	specifies the output TOPS file and the path to it
-q	overwrites the TOPS init string (Initialization String)
-j <input_simulation_file_location>	specifies an input simulation data file and the path to it
-s <output_simulation_file_location>	specifies an output simulation data file (*.struct) and the path to it
-t <output_simulation_file_location>	specifies an output simulation data file (*.txt) and the path to it
-w [<TOPS_version>]	sets the TOPS output version. Default is the current version.

Note: Table 10.5 lists only options available for multiple actions. For a complete list of all options specific for one action, please use the action argument followed by the -h option, e.g. gesop.exe -G -h

Example: Create an Initial Guess with Control Laws

The following command line example starts an Initial Guess generation task using the control laws specified in the scenario for the controls generation

```
%ASTOS%\gesop\bin\gesop.exe -G Initial_Guess -c "%Scenario_Path%"↲
    -m Data_Stream -lib "" -i "%Scenario_Path%\input.tops"↲
    -o "%Scenario_Path%\input.tops"↲
    -gs "%Scenario_Path%\integ\ini_guess.struct"↲
    -gt "%Scenario_Path%\integ\ini_guess.txt"
```

Table 10.6: Arguments used to create an initial guess with control laws in the example above

Argument	Description
`-G Initial_Guess`	creates an initial guess using the control laws defined in the scenario
`-c "%Scenario_Path%"`	set the working directory to the scenario .gtp folder
`-lib ""`	this example does not contain additional libraries
`-i "%Scenario_Path%/input.tops"`	specifies the input TOPS for the initial guess generation. A TOPS file is always required.
`-o "%Scenario_Path%/input.tops"`	adds state and control values of the initial guess generation to the `input.tops` file
`-gs "%Scenario_Path%/integ/ini_guess.struct"`	create a .struct simulation data file containing the results of the initial guess
`-gt "%Scenario_Path%/integ/ini_guess.txt"`	create a .txt file containing the results of the initial guess

Example: Simulate a Scenario

The following command line example starts a Simulation task a the scenario with an active **Optimization** feature

```
%ASTOS%\gesop\bin\gesop.exe -S Connected_Phases↵
    -c "%Scenario_Path%" -m Data_Stream -lib "" -i "%Scenario_Path%\input.tops"↵
    -t "%Scenario_Path%\integ\simulation.txt"↵
    -s "%Scenario_Path%\integ\simulation.struct"
```

Note: The scenario has to be initialized before running a simulation.

Run a Batch Mode

Note: In the current *ASTOS* version there is no support for a Batch Mode session to be run on the command line.

Results Export

Table 10.7 gives an overview of the supported result exports when *ASTOS* is used on the command line (see Section 10.4 for details).

Table 10.7: Support for the results export when ASTOS is run from the command line

Export Type	Supported by the gesop.exe interface	Supported by the start.bat/.sh interface
CCSDS	yes	yes

Export Type	**Supported by the gesop.exe interface**	**Supported by the start.bat./.sh interface**
Excel	no	yes
GESOP Structure	no	yes
Google Earth	yes	yes
MBS (Multibody Dynamics Simulation)	yes	yes
MySQL	no	yes
STK Animation	yes	yes
Text	yes	yes

11 Glossary

The following list contains a summary of definitions and acronyms often used in this manual. The items are listed in alphabetic order.

- **APS**
  ASTOS **P**roblem **S**et
  This is the main folder where normally several GTP folders are placed
- **ASTOS**
  ASTOS (**A**nalysis **S**imulation and **T**rajectory **O**ptimization **S**oftware for Space Applications) is a comprehensive environment plus an elaborate model library with several "applications", in such a way that the user does not have to do any coding. The complete scenario description is driven by a set of data, see chapter Introduction in Model Reference. *ASTOS* can be applied to several scenarios including satellites, launch and re-entry vehicles. The GUI takes the full responsibility of creating and modifying the required data files.
- **Auxiliary Function**
  These additional output functions are function of the independent variable and may reflect anything being neither a state nor a control.
- **Auxiliary Item**
  These are points, polygon or areas that identify objects fixed during the trajectory, e.g. impact position of a fragment or ground station position.
- **BER**
  Bit **E**rror **R**ate
- **BFGS**
  Broyden, **F**letcher, **G**oldfarb, **S**hanno: a quasi-Newton method to approximate the Hessian matrix.
- **CAMTOS**
  Collocation **a**nd **M**ultiple Shooting **T**rajectory **O**ptimization **S**oftware (see also Description of Methods in Optimization Theory and Description of Methods)
- **CCSDS**
  Consultative **C**ommittee for **S**pace **D**ata **S**ystems
- **CDF**
  Concurrent **D**esign **F**acility
- **CDMA**
  Code **D**ivision **M**ultiple **A**ccess
- **CGA**

Constrained **G**enetic **Al**gorithm (see also Description of Methods in Optimization Theory and Description of Methods)

- **Constraint**
  Constraints are defined for each phase. There exists initial boundary, path, final boundary and parameter constraints. The initial and final boundary constraints are evaluated at the initial and the final independent variable of a phase, respectively. The path constraint is evaluated at each constraint evaluation node except the initial and the final one. The parameter constraints are evaluated inside a phase with no reference to a certain independent variable value. All (active) constraints must be fulfilled during optimization.
- **Control**
  A control is a function of the independent variable. The optimizer is free to change the value of a control inside the specified bounds at each control evaluation node. There is no limitation on the number of controls an optimal control problem may consist of. Note, the control needs to be distinguished from optimizable parameters, like integer and real parameters.
- **Cost Function**
  The cost function is the objective of the optimal control problem and can be defined for each phase. *ASTOS* supports three different cost functions: initial cost, integrated cost (Lagrange cost term), and final cost (or terminal cost). The initial and final costs are Mayer cost terms and computed at the initial and the final value of the independent variable of a phase, respectively. The Lagrange cost is evaluated at each major grid node except the initial and the final one and integrated as a function of the independent variable.
- **C/N_0**
  Carrier-to-Noise Density Ratio (or CNR)
- **CNR**
  Carrier-to-Noise Ratio (see **C/N_0**)
- **DLL**
  Delay **L**ock **L**oop
- **Eb/N0**
  energy per bit to noise power spectral density ratio (normalized CNR)
- **EIRP**
  Equivalent **I**sotropically **R**adiated **P**ower
- **FilterSQP**
  Filter **S**equential **Q**uadratic **P**rogramming (see also Nonlinear Problem Solver)
- **FDMA**
  Frequency **D**ivision **M**ultiple **A**ccess
- **FLL**
  Frequency **L**ock **L**oop
- **GDOP**
  Geometric **D**ilution **O**f **P**recision
- **GESOP**
  GESOP (**G**raphical **E**nvironment for **S**imulation and **Op**timization) is a simulation and optimization environment to compute optimal trajectories for a variety of complex multi-phase optimal control problems. *GESOP* is the basic software for trajectory optimization.

It solves multi-phase trajectory optimization problems. In this basic version the user has to provide his own models.

- **GLONASS**
  GLObal **NA**vigation **S**atellite **S**ystem
- **GNSS**
  Global **N**avigation **S**atellite **S**ystem
- **GPS**
  Global **P**ositioning **S**ystem
- **GTP**
  GESOP **T**rajectory **P**roblem
- **G/T**
  Gain-to-noise **T**emperature
- **HDOP**
  Horizontal **D**ilution **O**f **P**recision
- **Independent Variable**
  All states and controls, as well as all other functions (e.g. auxiliary functions), are a function of the independent variable. Also the grids are defined as a function of the independent variable. Typically, for optimal control problems the Time is used as independent variable. Another example is to use an angle for e.g. multi revolution orbital transfer problems.
- **Integer Parameter**
  An integer parameter can be modified freely by the optimization algorithm inside the specified bounds, but only integer numbers are valid values. The value of an integer parameter is defined for a whole phase, but might change from one phase to the next phase. See also "Real Parameter".
- **ipfilter**
  interior **p**oint **filter** (see also Nonlinear Problem Solver).
- **Isp**
  It is a way to describe the efficiency of rocket and jet engines. It represents the force with respect to the amount of propellant used per unit time. It is usually expressed in seconds.
- **ITU**
  International **T**elecommunication **U**nion
- **MDEV**
  Modified Allan **DEV**iation
- **NLP**
  Non**l**inear **P**roblem
- **ODE**
  Ordinary **D**ifferential **E**quation
- **ODIN**
  Optimal **D**esign **IN**vestigation is a structural optimization software developed by MT-Aerospace; it enables the optimization of typical axis-symmetric structures by means of sizing the stiffening design concerning strength and stability while minimizing the structural mass of stage sub-components.
- **PDOP**
  Position **D**ilution **O**f **P**recision

- **Phase**
  A phase is a part of an optimal control problem where the optimal control problem setup is steady. In case of multi-phase problems the setup may change from phase to phase, but note, there are optimal control problems with only one phase (so called single-phase problems). Multi-phase problems are usually used to describe a changing environment, or to change the control, states, or parameters of the problem. Examples are changing equations of motion, additional rotational states (instead of translational states only), change of an environmental model (e.g. atmosphere) or the central body, etc.
- **Phase Connect (Condition)**
  A phase connect or phase connect condition determines how and to what extent a certain state or parameter value can vary from the end of a phase to beginning of a subsequent phase.
- **PLL**
  Phase **L**ock **L**oop
- **PROMIS**
  Parametrized T**r**ajectory **O**ptimization by Direct **M**ult**i**ple **S**hooting
  Multiple shooting transcription method that can use *SNOPT* or *SLLSQP* as solvers (see also Description of Methods in Optimization Theory and Description of Methods).
- **Problem and Problem Set**
  Throughout this manual often the noun **problem** is used. It is a short notation for "optimization problem" or formulation of a problem to optimize. In the *ASTOS* environment this designates a directory with a special structure containing all the information necessary for running the optimization, simulation and so on. Related problems (problems that share the same model) can be grouped in a directory to form a "**problem set**".
- **Real Parameter**
  A real parameter can be modified freely by the optimization algorithm inside the specified bounds. Each real number is valid as value and it is defined for a whole phase, but might change from one phase to another one. See also "Integer Parameter".
- **RHCP**
  Right **H**and **C**ircular **P**olarization
- **RPA**
  Rocket **P**ropulsion **A**nalysis
  It is a software for prediction of rocket engine performance at the conceptual and preliminary stages of design [33]. It has been integrated in *ASTOS* and can be used to provide the equilibrium calculation, mass estimation and engine dimension for throat design engines (see Throat Design Rocket in Model Reference).
- **SLLSQP**
  Sequential **L**inear **L**east **Sq**uares **P**rogramming
  Dense solver used by PROMIS (see also Nonlinear Problem Solver), it doesn't require an additional license feature.
- **SNLPMN**
  Sparce SQP **Non**l inear **P**rogram **Ma**i**n** is the sparce solver for *SOS* (see also Nonlinear Problem Solver).
- **SNR**
  Signal-to-**N**oise **R**atio or S/N

- **SNOPT**
  Sparse **N**onlinear **Opt**imizer
  Sparce solver licensed by Stanford University used by *CAMTOS* and *PROMIS* (see also Nonlinear Problem Solver).
- **SOS**
  Sparce **O**ptimization **S**uite
  Transcription method for huge problems (see also Description of Methods in Optimization Theory and Description of Methods).
- **SQP**
  Sequential **Q**uadratic **P**rogramming
- **State**
  A state is a function of the independent variable. The state is computed at each major grid node either analytically or numerically via integration of the state derivatives (right hand side).
- **TDOP**
  Time **D**ilution **O**f **P**recision
- **TEME**
  True **E**quator **M**ean **E**quinox
- **TIC**
  Truncated **I**deal **C**ontour
- **TLE**
  Two **L**ine **E**lement
- **TOPS**
  A **TOPS** (short for **T**rajectory **OP**timization **S**pecification) is a file which contains all the information to run the optimization and simulation programs: program parameters like optimization accuracy, a string to initialize the model and finally the grid data - how many phases there are, what phase times, which state and control vectors to use and so on.
- **TROPIC**
  Trajectory **Op**timization by D**i**rect **C**ollocation (see also Description of Methods in Optimization Theory and Description of Methods).
- **UERE**
  User **E**quivalent **R**ange **E**rror
- **VDOP**
  Vertical **D**ilution **O**f **P**recision
- **WGS**
  World **G**eodetic **S**ystem
- **WORHP**
  We **O**ptimize **R**eally **H**uge **P**roblems
  Sparce solver used by CAMTOS (see also Nonlinear Problem Solver).

12 How to Obtain Support

There are two official contact email addresses available to *ASTOS* users:

- service@astos.de: this address should be used for general support questions related to an efficient way of working with *ASTOS* or for commercial information regarding new features.
- user-feedback@astos.de: this address should be used to report bugs, suggest improvements and provide generic feedbacks.

Please do not hesitate to contact us, we guarantee a fast answer.

There are several groups of possible errors and several possibilities of support which also depend on the security classification of your scenario.

- **Error while editing input fields**:
  Due to the complexity of the scenario many input errors may generally occur. The goal of the GUI is to reduce them, to catch them and to guide the user through a solution process. The user can enter values and strings; one of the most common error is an invalid value (e.g. -3-4 or 12,24). In this case, a yellow background is present while editing and an error message window is opened once the modification is completed. The GUI discards the invalid input and sets the value to the previous (valid) one. Such kind of error is automatically corrected in the case of units, please refer to Section 2.10.3 for additional information about units.
- **Error while loading a scenario**:
  The *ASTOS* GUI is storing the scenario data in `XML` files to take over the (bug-prone) programming load from the user. Therefore the `XML` file should be touched only by the GUI to prevent introducing bugs. In principle thus, loading errors may only occur if the `XML` files are modified manually. In this case, an error is printed in the message panel, more information can be found in Section 2.1.
- ***ASTOS* GUI Java exceptions**:
  ASTOS GUI runtime errors should not occur but cannot be completely avoided due to the complexity of the dynamic graphical interface. Such errors open a *Java Exception* window which suggests sending the content to user-feedback@astos.de. All Java errors are written to `...\Astos Solutions\ASTOS\$version_number$\gesop_error.txt`, where `...` depends on the used operating system; in case of *Microsoft Windows* 7 and 10 the path is `C:\Users\$user_name$\AppData\Roaming`.
  Please support our effort in providing a bug-free software by sending the automatically generated email with a short explanation of the workflow which has led to the runtime error. If there is no possibility to send the automatically generated email, please send us the

`gesop_error.txt` file along with a short explanation of the workflow which has led to the runtime error.

- **Error during the model initialization**:
  The reason for such error lies in a faulty model definition of the *ASTOS* model files. First of all we need the error message from the *Execution Log* window, for additional details please see Section 2.3. If the problem cannot be solved, we would need parts of the `\model\astos` subfolder files or of the description interface for support. Normally no need specific information about the model (e.g. tabular data) is required.
- **Error during an initial guess or simulation**:
  These cause of these errors is similar to the one for model initialization errors. Additionally, the integration of the equation of motions or the execution of an analysis could have encountered some physical or mathematical problem. The information printed in the *Execution Log* window (Section 2.3) can help to identify the problem and the associated solution.
- **Error during an optimization**:
  Errors during the optimization can be very complex to identify and solve. The cause depends on the error and the model definition. First of all an optimization output from the *Execution Log* window is required to allow support. Later on, the `TOPS` file or also the complete scenario may have to be made available to identify the error. Some solvers provide an additional output file that can be kept with the appropriate flag in the *Preferences* window Section 2.9; additional details are provided in Monitoring Optimization Iterations in Optimization Theory and Description of Methods.

The following list groups possible support levels depending on the information that is provided to us and describes, if such information may contain confidential information. Please note, that any of our support is treated as confidential. Users obtain a non-disclosure agreement any time requested.

- **Copy of the ASTOS execution log**:
  The *Execution Log* window contains all output of your simulation, initial guess, optimization, result summary and all model error messages (see also Section 2.3). The result summary contains constraint values, cost function, real parameter and phase times, but no table data and normally also no mass information or other model details (see also Section 8.5). An optimizer output contains no model specific information (see also Monitoring Optimization Iterations in Optimization Theory and Description of Methods) as well. It contains only information of the parameterized problem, which reveals normally nothing about the model. The user is able to edit the log and to delete some information for security reasons. Such information should not be critical for the error identification. The complete log can be exported to a file using the save button, or some selected lines can be copied via *Strg+C*.
- **TOPS file**:
  The `TOPS` file (e.g. input.tops) allows us to load your scenario and analyse it. It contains only the parameterized problem similar to the optimizer output but we have additional access to some state and control values. In case of optimization scenarios this may help us to understand the issue.
- **Simulation file**:

The simulation.struct file (in the `\integ` subfolder) allows us to plot all auxiliary functions of your scenario. In case any model information is included (e.g. aerodynamic coefficients as function of the time or mass flow and thrust), such information can be classified, since some errors can only be detected with the simulation files.

- **Model files**:
  The files in the `\model\astos` subfolder contain the model information. These files together with `input.tops` allow us to simulate or optimize your scenario, i.e. we have all information about your scenario and we are able to modify it. In few cases this is the only way to identify the error.
- **Complete .gtp folder**:
  Please compress the complete `.gtp` folder either manually or by selecting in the ribbon *Application menu→Compress Scenario*.

References

[1] Betts J.T., "SOS User's Guide - Release 2012.08", published by AMA, Issaquah (US) 2012.

[2] Betts J.T., "Practical Methods for Optimal Control Using Nonlinear Programming", Advances in Design and Control, Society for Industrial and Applied Mathematics (SIAM), 2001.

[3] M. C. Bartholomew-Biggs and L. C. Dixon and S. E. Hersom and Z. A. Maany and M. Hechler (1987): From High Thrust to Low Thrust: Application of Advanced Optimization Methods to Mission Analysis, ESA Journal, No. 11.

[4] C. Jänsch and D. Kraft and K. Schnepper and K. H. Well (1989): Survey on Trajectory Optimization Methods, ESA/ESTEC Noordwijk/NL, ALTOS TR1, Contract No. 8046/88/NL/MAC.

[5] P. E. Gill and W. Murray and M. A. Saunders (1995): An SQP Algorithm for Large Sparse Optimization, Report NA 95-0, Department of Mathematics, UCSD.

[6] Jänsch C., Schnepper. K., Well. K.H. (1994): Multiphase trajectory optimization methods with applications to hypersonic vehicles , Applied mathematics in aerospace science and engineering (Miele and Salvetti, Eds.), Chapter 8, Plenum Press, New York.

[7] P. E. Gill and W. Murray and M.H. Wright (1981): Practical Optimization, Academic Press, New York.

[8] Wolfram, S. Mathematica - A System for Doing Mathematics by Computer, Addison-Wesley, Reading, Mass., 1988.

[9] Vallado D. A., *Fundamentals of Astrodynamics and Applications*, 3rd Edition, Space Technology Library, Springer, New York, 2007.

[10] Consultative Commitee for Space Data Systems, *Recommendation for Space Data System Standards, Time Code Formats*, CCSDS 301.0-B-3, Blue Book, Issue 3, CCSDS, January 2002.

[11] Seidelmann P. K., Abalakin V. K., Bursa M., Davies M. E., de Bergh C., Lieske J. H., Oberst J., Simon J. L., Standish E. M., Stooke P. and Thomas P. C., *Report of the IAU/IAG Working Group on Cartographic Coordinates and Rotational Elements of the Planets and Satellites: 2000*, Celestial Mechanics and Dynamical Astronomy, Vol. 82, No. 1, January 2002.

[12] U.S. Standard Atmosphere, 1962, NOAA, NASA, US Air Force, Washington, D.C., 1962.

[13] U.S. Standard Atmosphere, 1976, NOAA, NASA, US Air Force, Washington, D.C., 1976; ANSI/AIAA G-003B-2004, Guide to Reference and Standard Atmosphere Models, 2004.

[14] Shau G.-C. *Der Einfluß flugmechanischer Parameter auf die Aufstiegsbahn von horizontalstartenden Raumtransportern bei gleichzeitiger Bahn- und Stufungsoptimierung*, German Aerospace Research Establishment, DLR Rept. 73-86, Oberpfaffenhofen, Germany, 1973.

[15] http://ccmc.gsfc.nasa.gov/modelweb/atmos/msis.html

[16] http://ccmc.gsfc.nasa.gov/modelweb/atmos/msise.html

[17] http://ccmc.gsfc.nasa.gov/modelweb/atmos/nrlmsise00.html

[18] Bowman B., *Jacchia-Bowman Thermospheric Density Model*, retrieved 2009 10 01, from http://sol.spacenvironment.net/~JB2006/introduction.html

[19] Justus C.G. et al., *The NASA/MSFC Global Reference Atmospheric Model - 1995 Version (GRAM-95)*, NASA Technical Memorandum 4715.

[20] Justus C.G., Johnson D.K., *The NASA/MSFC Global Reference Atmospheric Model - 1999 Version (GRAM-99)*, NASA/TM-1999-209630.

[21] Mars Climate Database official web site - http://www-mars.lmd.jussieu.fr/

[22] CEA, Chemical Equilibrium with Applications, NASA Glenn Research Center, http://www.grc.nasa.gov/WWW/CEAWeb/

[23] Cash J. *Solver MEBDFDAE* - http://www.dm.uniba.it/~testset/report/mebdfdae.pdf

[24] User's Guide to WORHP 1.2, 2013, Steinbeis Forschungszentrum Optimierung, Steuerung und Regelung.

[25] Rosser J., Barkley, *Mathematical Theory of Rocket Flight*, McGraw-Hill Book Company, New York and London, 1947.

[26] Buhl W. *Private Communications*, Daimler-Benz Aerospace Co., Ottobrunn, Germany, Aug. 1995.

[27] Roenneke A. J. *ALTOS-2 Reentry Problem Set*, Advanced Launch and Reentry Trajectory Optimization Software Documentation, Technical Report, Institute of

Flight Mechanics and Control, University of Stuttgart, Rept. IFR TR 95-06, Jan. 1995.

[28] Amblard P. *Preliminary Requirements for Stabilizing Function Hermes R-Shape*, Avions Marcel Dassault Breguet Aviation Espace, Rept. H-NT-1C-1004-AMD, Jan. 1989.

[29] Roenneke A. J. *Aerodynamic Model of a High-Lift Reentry Vehicle for Controller Design and Simulation*, Institute of Flight Mechanics and Control, University of Stuttgart, Rept. IFR TR 94-04, Nov. 1994.

[30] Vignials I. *Hermes R-Shape Aerodynamic Data Base*, Avions Marcel Dassault Breguet Aviation Espace, Rept. H-NT-1-1074-AMD, Nov. 1988.

[31] Schöttle U. and Rahn U. *Fahrzeugmodelle für Sensitivitätsstudien konventioneller Trägerraketen*, Institute of Space Systems, University of Stuttgart, Rept. IRS 95-IB-11, Aug. 1995.

[32] Akin, David L., *Principles of Space Systems Design*, Mass Estimating Relationships, Course Material, University of Maryland, 2004.

[33] *Rocket Propulsion Analysis Software*, http://propulsion-analysis.com/

[34] Roenneke A.J., "Ein bordautonomes Verfahren zur Flugführung und Regelung von Rückkehrfahrzeugen", Dissertation, Institute of Flight Mechanics and Control, University of Stuttgart, 2001 (in German)

[35] Roenneke A.J., "Adaptive On-Board Guidance for Entry Vehicles", AIAA 2001-4048, AIAA Guidance, Navigation, and Control Conference, Montrèal, Québec, Canada, 6-9 August 2001

[36] International Geomagnetic Reference Field, http://www.ngdc.noaa.gov/IAGA/vmod/igrf_old_models.html

[37] CCSDS - Attitude Data Messages, Recommended Standard CCSDS 504.0-B-1, Blue Book, May 2008

[38] CCSDS - Orbit Data Messages, Recommended Standard CCSDS 502.0-B-2, Blue Book, November 2009

[39] Gallais P., *Atmospheric Re-Entry Vehicle Mechanics*, Springer, Berlin Heidelberg 2007.

[40] Astos Solutions GmbH, *DCAP User's Manual*, 2020.

[41] Schlueter M., Munetomo M., *MIDACO User Guide*, Technical Report, Hokkaido University, Japan (HUSCAP) (2013).

[42] Gill, P.E., Murray, W., Saunders, A., "SNOPT 6 Runtime User Software User Manual", based on [43] published by Astos Solutions, 2009.

[43] Gill, P.E., Murray, W., Saunders, A., "User's Guide For Snopt Version 6, A Fortran Package for Large-Scale Nonlinear Programming", December 2002.

[44] *Satsearch*, https://satsearch.co/

[45] H. J. Landau, "Sampling, data transmission, and the Nyquist rate," in Proceedings of the IEEE, vol. 55, no. 10, pp. 1701-1706, Oct. 1967