GENESYS™ 2023

MATLAB® Constraint Solver

Zuken Vitech Inc.
2270 Kraft Drive, Suite 1600
Blacksburg, Virginia 24060
540.951.3322 | FAX: 540.951.8222
Customer Support: support@vitechcorp.com
www.vitechcorp.com

Publication Date: June 2023

TABLE OF CONTENTS

LIST OF FIGURES

CUSTOMER RESOURCE OPTIONS

Supporting users throughout their entire journey of learning model-based systems engineering (MBSE) is central to Vitech's mission. For users looking for additional resources outside of this document, please refer to the links below. Alternatively, all links may be found at www.vitechcorp.com/online-resources/.

Webinars

Immense, on-demand library of webinar recordings, including systems engineering industry and tool-specific content.

Screencasts

Short videos to guide users through installation and usage of GENESYS.

A Primer for Model-Based Systems Engineering

Our free eBook and our most popular resource for new and experienced practitioners alike.

Help Files

Searchable online access to GENESYS help files.

Technical Papers

Library of technical and white papers for download, authored by Vitech systems engineers.

Technical Support

Frequently Asked Questions (FAQ), support-ticket web form, and information regarding email, phone, and chat support options.

PREFACE

The MATLAB Constraint Solver provides the ability to solve a set of parametric equations constraining the design of a system. The MATLAB Constraint Solver is integrated into the GENESYS Pro version and uses MATLAB Version 2015b or higher, to solve a selected set of **ConstraintDefinitions** determined by the user. The MATLAB Constraint Solver converts project relationships, parameter values, and attribute values into scripts that can be directly executed against the MATLAB engine without having to leave the GENESYS interface. Following script execution, project parameter values can be updated based on the MATLAB solution. An overview of the MATLAB Constraint Solver is shown in the following graphic.

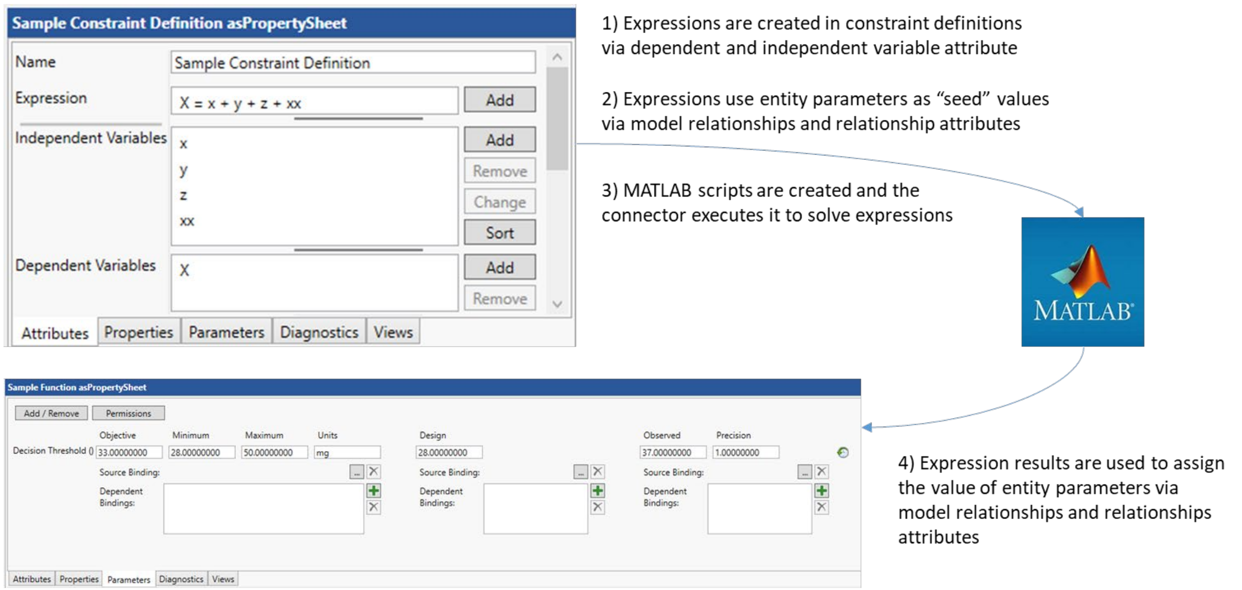

Figure 1 – MATLAB Constraint Solver Concept Graphic

This guide describes the processes used to populate and model **ConstraintDefinition** entities, declare and populate parameters in **Component** and **ConstraintDefinition** entities, and solve a selected set of **ConstraintDefinition** expressions using the MATLAB Constraint Solver.

This guide is intended to augment the Model-Based Systems Engineering (MBSE) with GENESYS training course and the reference material provided with GENESYS. The ultimate goal of this guide is to expose the user to the MATLAB Constraint Solver and thereby extend the use and application of GENESYS for system design, development, and project management associated with the system development project.

The following additional resources are available for use with this guide:

- For descriptions of GENESYS, including database classes and folders, different views, diagram notation, and the mechanics of entering data into GENESYS, the reader is referred to the GENESYS Help and Documentation guide.
- For the definition of schema terms, the reader is referred to the GENESYS schema, which contains descriptions for each schema entity, associated attributes, and parameters.

For application of GENESYS to system and architecture design, the reader is referred to the GENESYS System Definition Guide (SDG) and Architecture Definition Guide (ADG) each of which is provided in the set of GENESYS documentation supplied when GENESYS is installed on a computer workstation or available on our website.

CONSTRAINT DEFINITION IN GENESYS

1.1. Constraint Block Definition Diagram

Utilizing the MATLAB Constraint Solver begins with defining constraints in the system design. The **ConstraintDefinition** captures the definition of parametric constraints as an expression (or equation) and identifies the independent variables and the dependent variable as attributes of the entity. The **ConstraintDefinition** is used in development of both the Constraint Block Definition Diagram and the Parametric Diagram in GENESYS.

Generally, a **ConstraintDefinition** *constrains* a **Component** in the architecture. Relevant schema relations for the **ConstraintDefinition** are shown in the following diagram.

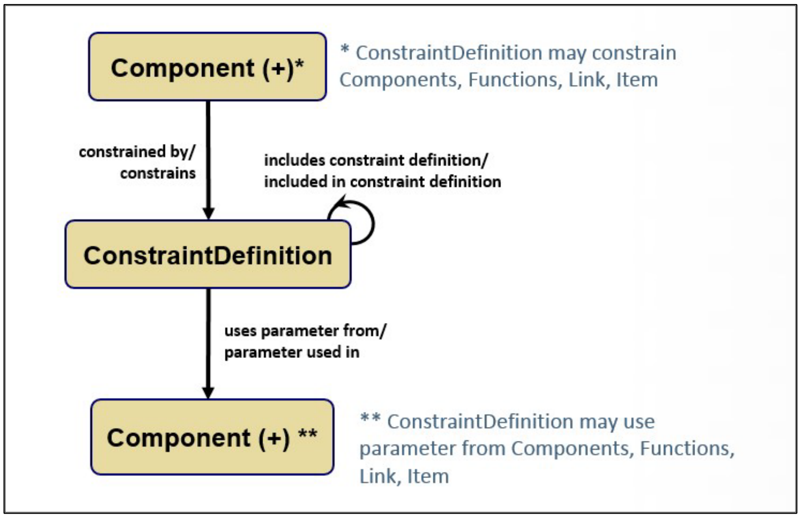

Figure 2 – Constraint Definition Schema Relations

The **ConstraintDefinition** property page contains a unique set of attributes as shown in the diagram below.

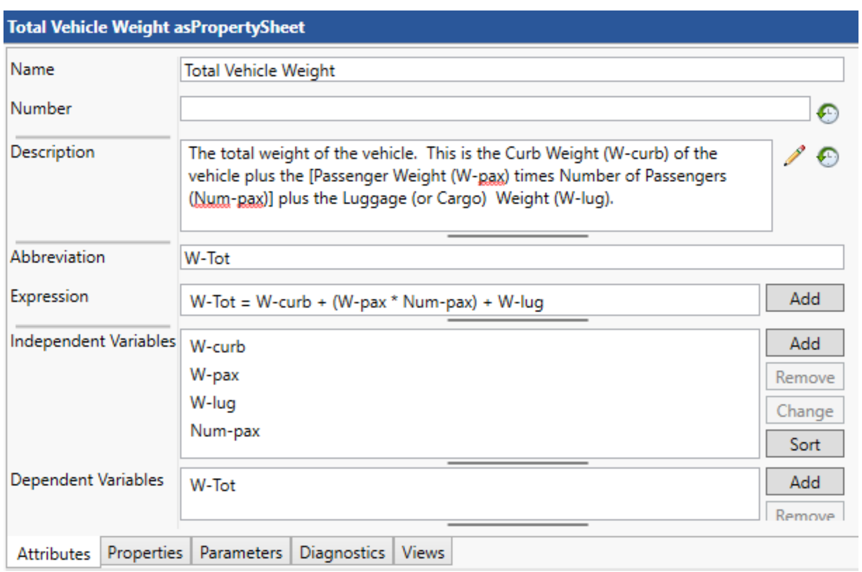

Figure 3 – Constraint Definition Property Page

In the property page the Expression attribute is used to capture the constraint's mathematical equation(s) using independent variables and dependent variables.

From the perspective of a **Component** in the architecture, the *constrained by* relation is used to create the Constraint Block Definition Diagram (Constraint BDD). A **ConstraintDefinition** entity can *use parameter from* another **Component**. An example Constraint BDD for **Component** "New Vehicle" is shown below.

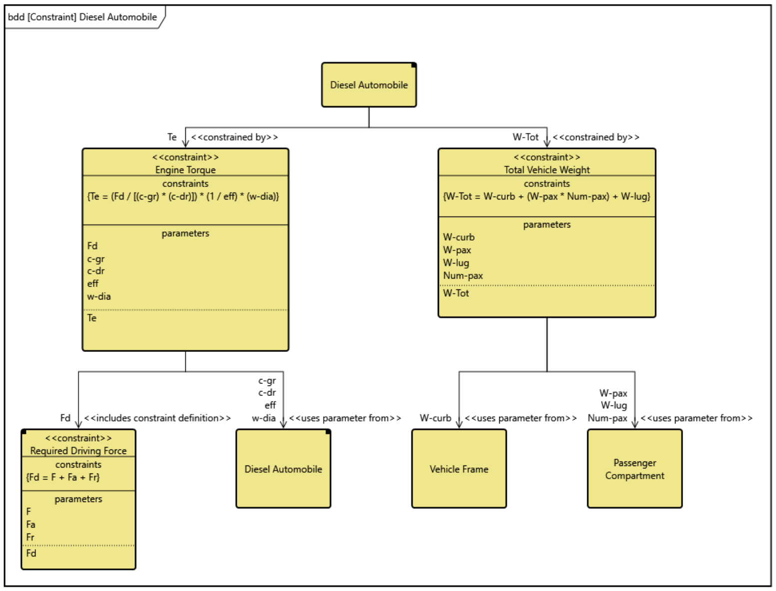

Figure 4 – New Vehicle Constraint bdd

PARAMETER DEFINITION IN GENESYS

1.2. Defining Parameters

The use of **Component** parameters is essential when managing and solving scripts derived from **ConstraintDefinition** models using the MATLAB Constraint Solver. The Parameters tab has been updated in GENESYS to expand the ability to identify and manage values.

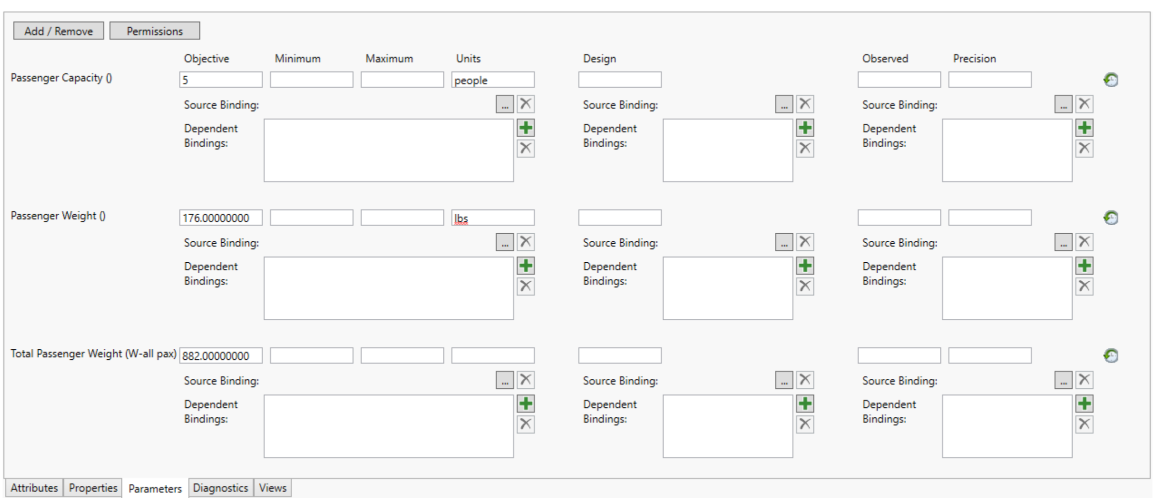

Figure 5 – Parameters Tab

The user can now define Objective, Minimum, Maximum, Design, Observed, and Precision parameter values. The Objective value is used to capture the value nominally defined through a **Requirement**. The

Design value is used to capture the calculated design value (the value determined using the MATLAB Constraint Solver). The Observed value is used to capture the value for the “as built” system.

Parameters can be copied from those defined in other classes in the project. Additionally, the definition page includes the ability to define an abbreviation for the parameter. The parameter definition dialog box is provided in the following figure.

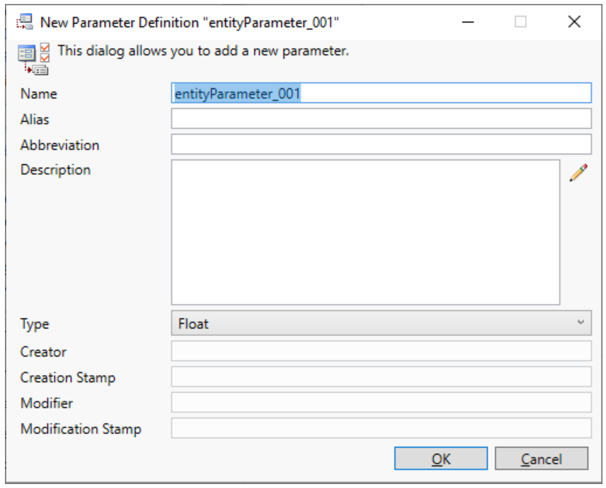

Figure 6 – Updated Parameter Definition

1.3. Creating ConstraintDefinitions

As the system design evolves from the design requirement architecture through the functional architecture, the design team will allocate requirement constraints (and their related parameters) to components in the physical architecture. The design team will also identify constraint definitions for individual components in the architecture.

A typical example is system weight. The overall system weight is a parameter that is tracked in the design. The overall system weight is a summation (or roll-up) for the weight of many individual components in the architecture. Therefore, we can define the system design weight as a constraint definition with an expression which adds the weight of several lower level components. Below is a constraint definition defining the “Total Vehicle Weight” for an automobile design.

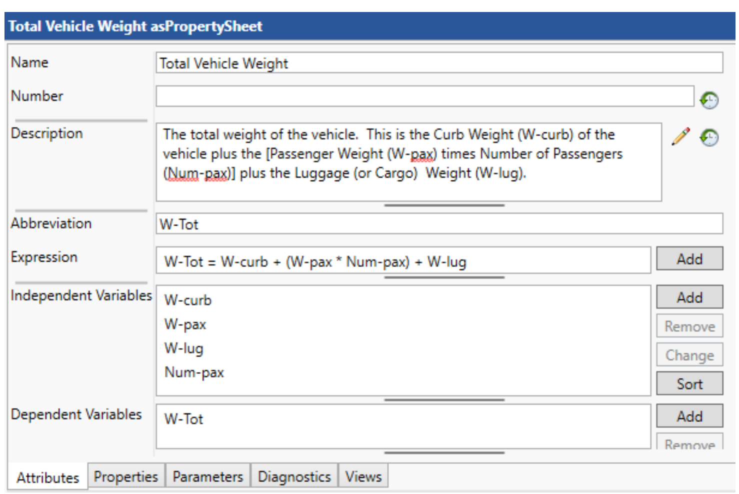

Figure 7 – Total Vehicle Weight Attributes

Once a **ConstraintDefinition** has been defined in the property sheet, the Independent Variables need to be associated with a parameter in the design. This is accomplished by relating the **ConstraintDefinition** with another entity in the architecture by establishing a *uses parameter from* or *includes constraint definition* relation. Each of these relations have a “Mappings” relationship attribute. The Mappings relationship attribute associates an individual Independent Variable with a parameter (*uses parameter from*) or a Dependent Variable (*includes constraint definition*) from the target entity. The Independent Variable will be seeded with the value of the parameter/dependent variable when the expression is evaluated using the MATLAB Constraint Solver.

The Dependent Variable constrains an entity in the design. This is established using the *constrains* relation and the “Mappings” relation attribute. The value of the dependent variable is calculated in the solution, and this value is placed in the Design attribute for parameter mapped to the dependent variable.

Using the example from the “Total Vehicle Weight” entity (shown in the prior diagram), the Independent Variable declared as “W-curb” takes its value from the parameter “curbWeight” which has been defined and associated with the **Component** “SYS.1 Vehicle Frame”. Independent Variables take their value from the Design Parameter value as a priority. If the Design value is nil, then the value is taken from the Objective value.

The Dependent Variable “W-Tot” *constrains* the **Component** “SYS New Vehicle”, and the value (which will be calculated by the MATLAB Constraint Solver) will be associated with the “totalVehicleWeight” parameter.

The relations and configurations of “Mappings” relationship attributes for “Total Vehicle Weight” are shown in the Targets & Attributes section of an entity property page. An example is shown below.

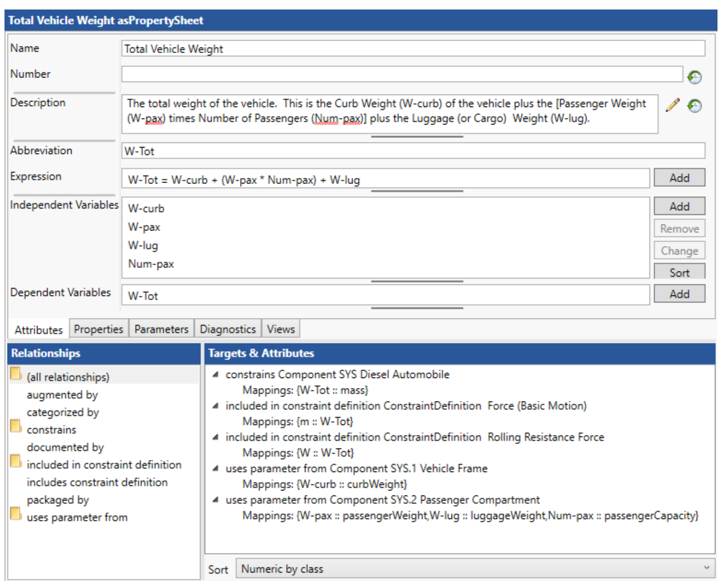

Figure 8 – Total Vehicle Weight with Mappings

The relation and its attributes are also displayed graphically in the parametric diagram for the **ConstraintDefinition**. The preferred way to set the "Mappings" relationship attribute is by using the Edit Mappings command on the parametric diagram when you have a line selected.

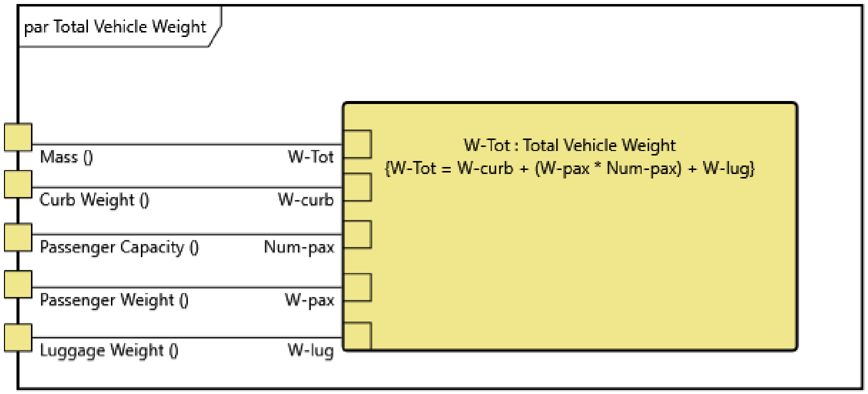

Figure 9 – Total Vehicle Weight Parametric Block Diagram

SOLVING EXPRESSIONS USING THE MATLAB CONSTRAINT SOLVER

1.4. MATLAB Constraint Solver

To execute the MATLAB Constraint Solver, select the ***Constraint Solver*** 

icon in the *MATLAB* section of the ***Utilities*** ribbon in GENESYS.

Figure 10 – Constraint Solver Command

When the Constraint Solver command is selected, a MATLAB Constraint Solver dialog box is opened in GENESYS. This is the first step in a series of dialog boxes that walk the user through the process of using the constraint solver. The first dialog box allows the user to select one or more **ConstraintDefinitions** that have been collected in a folder, package, or contained in a diagram.

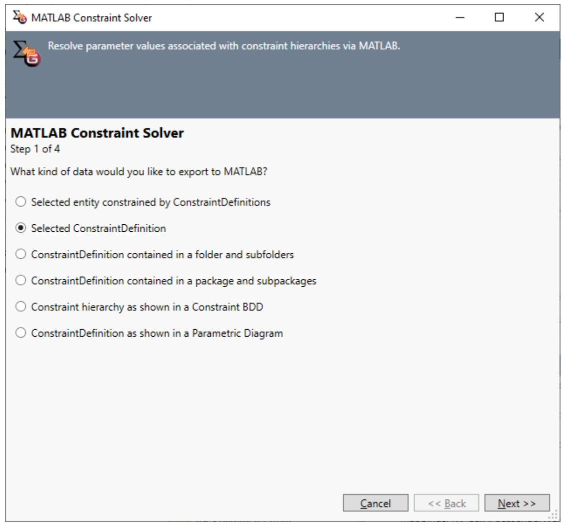

Figure 11 – Constraint Solver - 1st Dialog Box

Depending on the selection made in the first dialog box, the user will be directed to a second dialog box to select a particular **ConstraintDefinition**, folder, or package. The example below shows the selection options taken when using the “Selected ConstraintDefinition” option from the 1st dialog box.

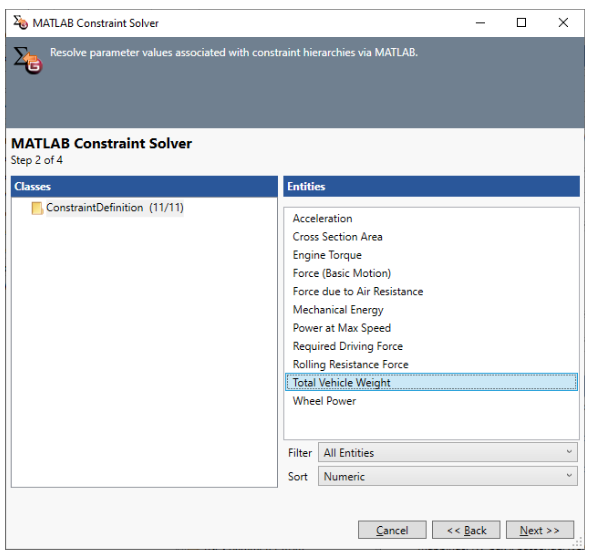

Figure 12 – MATLAB Constraint Solver - 2nd Dialog Box

In the example above, the user is selecting the **ConstraintDefinition** “Total Vehicle Weight” as the entity to be solved.

Once a selection is made, selecting “Next” brings up the “MATLAB script box” (the 3rd dialog box in the execution sequence). This box shows the parameters mapped to the **ConstraintDefinition**, as well as the expression and dependent variable mappings that will be used to create the script to be solved by MATLAB.

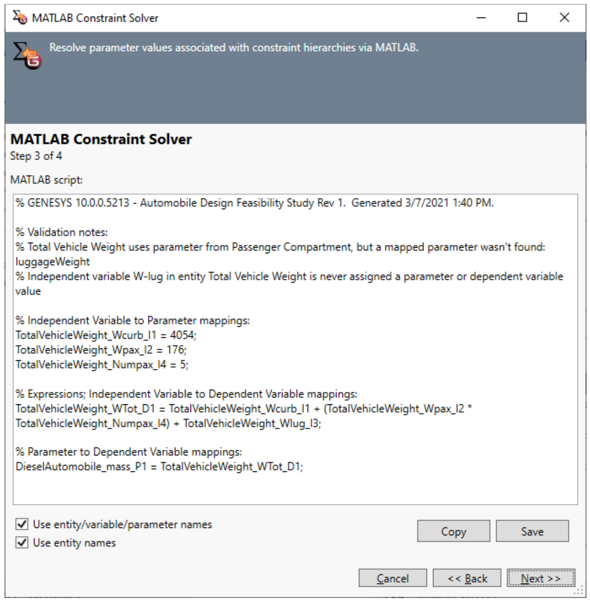

Figure 13 – MATLAB Script Dialog Box

The user can review the parameters, parameter values, and the expression and dependent variables being used prior to solving the script. The script can be saved as a text file if desired. When ready to complete the solution, the user will select the “Next” button to execute the MATLAB Constraint Solver.

Selecting the “Next” button will change the dialog box to show the MATLAB script results with a solution. The script box will indicate “(pending script execution)”, as shown below.

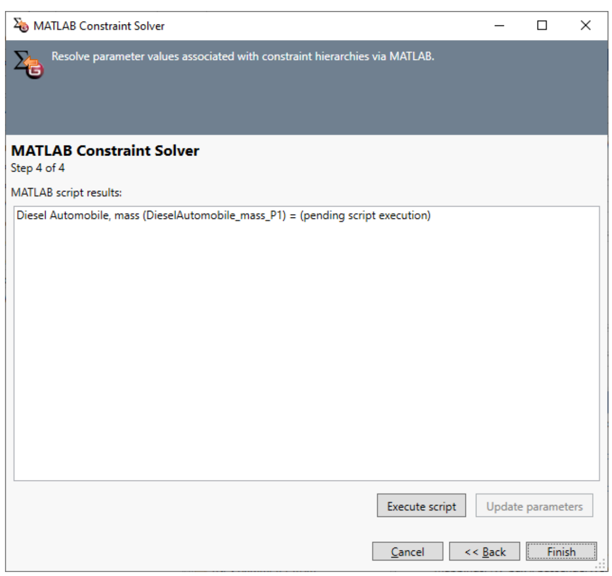

Figure 14 – MATLAB Constraint Solver - Pending Execution

Selecting the “Execute script” button from this page will execute the scripts against MATLAB’s solver engine and calculate the value for the dependent parameter(s) in the **ConstraintDefinition**.

After execution the “solved” parameter value is shown below.

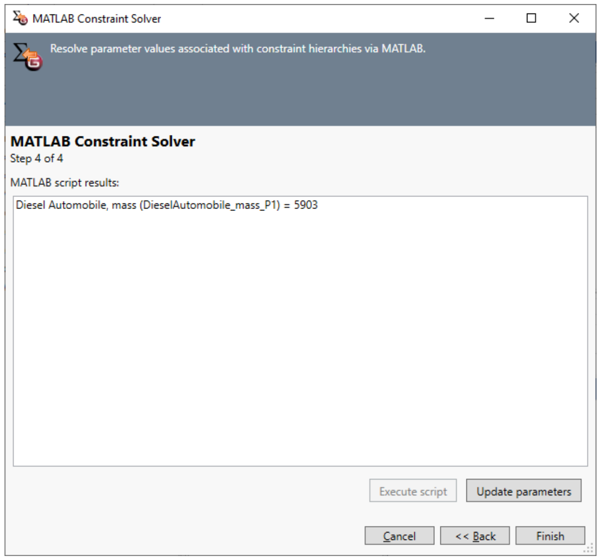

Figure 15 – MATLAB Constraint Solver Results Dialog Box

Following script execution, the “Update parameters” button can be used to place the parametric value into the appropriate “Design” column for the respective parameter. If a parametric value has a new (or updated) value, the dialog box will indicate that the parameter has been updated, as shown below.

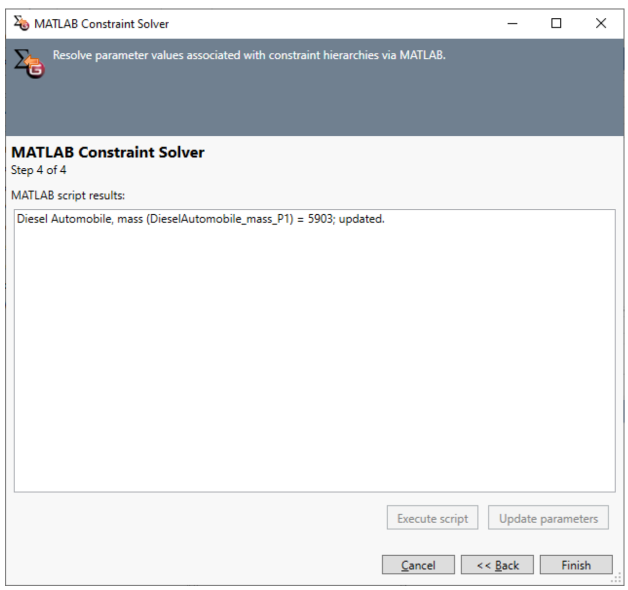

Figure 16 – MATLAB Solution - Updating Parameters

If the user then navigates to the updated parameter, the original value and the updated value (the value that has changed based on the MATLAB Constraint Solver solution) will be shown.

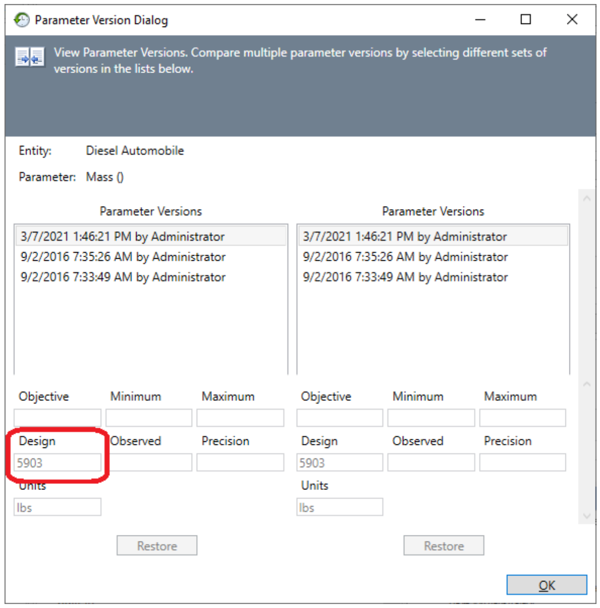

Figure 17 – Parameter Value Version

2270 Kraft Drive, Suite 1600
Blacksburg, Virginia 24060
540.951.3322 | FAX: 540.951.8222
Customer Support: support@vitechcorp.com
www.vitechcorp.com