General Mission Analysis Tool (GMAT)

User Guide

General Mission Analysis Tool (GMAT): User Guide

Table of Contents

Preface

The GMAT User's Guide contains material for new and experienced users and is organized into the following sections:

- Introduction
- Creating Your First Mission
- How-tos
- Tutorials
- Reference Guide

Introduction

The Introduction section contains two major parts: *Introduction to GMAT* and *Getting Started*.

The *Introduction to GMAT* section contains a brief project and software overview and discusses project status, licensing, and contributors.

The *Getting Started* section describes how to install and start GMAT, presents an overview of the user interfaces, and provides information on configuring your system.

> **Note**
>
> We consider the the section called "User Interfaces Overview" essential reading. If you read nothing else, at least read this section as it will explain the basic philosophy and rules of GMAT's user interfaces.

Creating Your First Mission

The Creating Your First Mission section walks you step-by-step through a sample mission, including creating a spacecraft, a propagator, and an `OrbitView` graphical display, and propagating the spacecraft to orbit perigee.

How-tos

The How-tos section contains many short articles that each describe a single area of functionality. The purpose is of the how-to documentation is to show you how to use a specific feature in an analysis context, and these articles often start from the default mission that is loaded when you start GMAT. A how-to article is designed to take about five minutes to teach you how to perform a specific task.

Tutorials

The Tutorials section describes how to use GMAT for end-to-end analysis. Tutorials are designed to teach you how to use GMAT in the context of performing a real-world analysis, and are intended to take between 30 minutes and one day to complete. Each tutorial as a difficulty level, an approximate duration, and potentially a number of prerequisites, all of which are listed in its introduction.

Reference Guide

The Reference Guide contains individual topics that describe each of GMAT's resources and commands in detail, including its syntax, options, variable ranges and data types, defaults, and expected behavior.

Part I. Introduction

Table of Contents

Introduction to GMAT

GMAT is an open source trajectory design and optimization system developed by NASA and private industry. It is developed in an open source process to maximize technology transfer and permit anyone to develop and validate new algorithms and to enable those new algorithms to quickly transition into the high fidelity core.

GMAT is designed to model and optimize spacecraft trajectories in flight regimes ranging from low Earth orbit to lunar, interplanetary, and other deep space missions. The system supports constrained and unconstrained trajectory optimization and built-in features make defining cost and constraint functions trivial. GMAT also contains initial value solvers (propagators) and boundary value solvers and efficiently propagates spacecraft either singly or as coupled sets. GMAT's propagators naturally synchronize the epochs of multiple vehicles and shorten run times by avoiding fixed step integration and interpolation.

Users can interact with GMAT using either a graphical user interface (GUI) or a custom scripting language modeled after the syntax used in The MathWorks' MATLAB® system. All of the system elements can be expressed through either interface, and users can convert between the two in either direction.

Analysts model space missions in GMAT by first creating and configuring resources such as spacecraft, propagators, optimizers, and data files. These resources are then used in a Mission Sequence to model the trajectory of the spacecraft and simulate mission events. The mission sequence supports commands such as nonlinear constraints, minimization, propagatation, GMAT or MATLAB functions, inline equations, and script events.

GMAT can display trajectories in a realistic three-dimensional view, plot parameters against one another, and save parameters to files for later processing. The trajectory and plot capabilities are fully interactive, plotting data as a mission is run and allowing users to zoom into regions of interest. Trajectories and data can be viewed in any coordinate system defined in GMAT, and GMAT allows users to rotate the view and set the focus to any object in the display. The trajectory view can be animated so users can watch the evolution of the trajectory over time.

Licensing

GMAT is licensed under the NASA Open Source Agreement v1.3. The license text is contained in the file `License.txt` in root directory of the GMAT distribution, and is listed in Appendix A.

Platforms

GMAT is implemented to run on Windows, Linux and Macintosh platforms, using the wxWidgets cross platform UI toolkit, and can be built using either Microsoft Visual Studio or the GNU Compiler Collection (GCC). GMAT is written in ANSI standard C++ (approximately 380,000 non-comment source lines of code) using an object-oriented methodology, with a rich class structure designed to make new features simple to incorporate. On Windows and Linux, GMAT does not call any operating-system-unique functions or methods. Calls to the operating system are standard calls for reading and writing data files and for writing data to the screen. On the Mac, GMAT makes a call to the operating system to open X11, which is required to run MATLAB on the Mac.

User Interfaces

GMAT has several user interfaces. The interactive graphical user interface is introduced in more detail in later sections. The script interface is textual and also allows the user to configure and execute all aspects of GMAT. There also a secondary MATLAB interface that allows for running the system via calls from MATLAB to GMAT and allows GMAT to call MATLAB functions from within the GMAT command sequence. A low-level C API is also currently under development.

Status

While GMAT has undergone extensive testing and is mature software, at the present time we consider the software to be in beta form on Windows and alpha on Linux and Mac. GMAT is not yet sufficiently verified to be used as a primary operational analysis system. It has been used to optimize maneuvers for flight projects such as NASA's LCROSS and ARTEMIS missions, and the Lunar Reconnaissance Orbiter, and for optimization and analysis for the OSIRIS and MMS missions. However, for flight planning, we independently verify solutions generated in GMAT in the primary operational system.

The GMAT team is currently working on several activities including maintenance, bug fixes, and testing, along with selected new functionality.

Contributors

The Navigation and Mission Design Branch at NASA's Goddard Space Flight Center performs project management activities and is involved in most phases of the development process including requirements, algorithms, design, and testing. The Ground Software Systems Branch performs design, implementation, and integration testing. The Flight Software Branch contributes to design and implementation. GMAT contributors include volunteers and those paid for services they provide. We welcome new contributors to the project, either as users providing feedback about the features of the system, or as developers interested in contributing to the implementation of the system. Current and past contributors include:

- Thinking Systems, Inc. (system architecture and all aspects of development;
- Air Force Research Lab (all aspects of development)
- a.i. solutions (testing);
- Boeing (algorithms and testing);
- The Schafer Corporation (all aspects of development);
- Honeywell Technology Solutions (testing);
- Computer Sciences Corporation (requirements);

The NASA Jet Propulsion Laboratory (JPL) has provided funding for integration of the SPICE toolkit into GMAT. Additionally , the European Space Agency's (ESA) Advanced Concepts team has developed optimizer plug-ins for the Non-Linear Programming (NLP) solvers SNOPT (Sparse Nonlinear OPTimizer) and IPOPT (Interior Point OPTimizer).

Getting Started

Installation

Installers and files for all platforms are located on the GMAT SourceForge page at `https://sourceforge.net/projects/gmat`. GMAT releases are listed in chronological order with the most recent release at the top of the list. As of this writing the latest version is R2011a, released April 29, 2011.

Installing on Windows

The GMAT windows distribution contains an installer that will install and configure GMAT for you automatically. By default GMAT will be installed into your Application Data folder, and a shortcut will be placed in the Programs menu.

Installing on Mac

GMAT for Mac is released as a compressed archive. The archive can be uncompressed and installed by either double-clicking on it, or by running the following command in a Terminal window:

```
tar -zxf gmat-snowleopard-x86-R2011a-Alpha.tar.gz
```

Either method extracts the GMAT system into a folder named `GMAT-R2011a`; this is the GMAT root directory. You may move this folder in its entirety to the Applications folder or your desired installation directory.

Installing on Linux

GMAT for Linux is released as a compressed archive. The archive can be uncompressed by running the following command:

```
tar -zxf gmat-linux-x64-R2011a-Alpha.tar.gz
```

This command extracts the GMAT system into a folder named `GMAT-R2011a`; this is the GMAT root directory. Inside of that folder you will find two application launchers, `RunGmat.sh` and `RunGmatLauncher.sh`. `RunGmatLauncher.sh` assumes that GMAT is installed in the user's home directory; update that location if you installed GMAT to a different location. The Linux application is launched by running one of these two shell scripts. Each script file sets GMAT's load library path, then launches the application. If you would like to create a desktop shortcut to launch GMAT, set your launcher to run `RunGmatLauncher.sh`. You might want to set the icon for the launcher as well; GMAT's icon images are located in the `GMAT-R2011a/data/graphics/icons` directory.

Starting and Quitting GMAT

Starting a GMAT Session

On Microsoft Windows platforms there are several ways to start a GMAT session. If you used the GMAT installer, you can click the `GMAT R2011a` item in the Programs menu. If you installed GMAT from a zip file, or by compiling the system, locate the bin folder in the GMAT root directory and double-click `GMAT.exe`.

On Mac, use the finder to open the `bin` folder located in your GMAT root directory and open the `GMAT` application. Alternatively, open a Terminal window, change to your installation directory, then type the command "**`open GMAT.app`**". Once GMAT is open, you can set it up to remain in the dock by clicking its dock icon, clicking Options, then Keep in Dock. This allows you to open GMAT in the future simply by clicking its dock icon.

Quitting a GMAT Session

To end a GMAT session on Windows or Linux, in the menu bar, click File, then click Exit. On Mac, in the menu bar, click **GMAT**, then click Quit GMAT, or type **Command**+**Q**.

Running the GMAT Demos

The GMAT distribution includes more than 30 sample missions. These samples show how to apply GMAT to problems ranging from the Hohmann transfer to libration point station-keeping to trajectory optimization. To locate and run a sample mission:

1. Open GMAT.
2. On the toolbar click Open.
3. Navigate to the `samples` folder located in the GMAT root directory.
4. Double-click a script file of your choice.
5. Click Run.

To run optimization missions, you need MATLAB®, and the MATLAB® Optimization Toolbox and/or the VF13ad plugin based on software in the Harwell Subroutine Library. These are proprietary libraries and are not distributed with GMAT. MATLAB® is not yet fully supported in the Mac and Linux GMAT releases, and therefore you cannot run optimization missions that use MATLAB's fmincon optimizer in the current Mac and Linux builds.

User Interfaces Overview

GMAT contains several user interfaces to design and execute your mission. The two primary interfaces are the graphical user interface (GUI) and the script interface. Each of these interfaces are interchangeable and support most functionality available in GMAT. When you work in the script interface, you are working in GMAT's custom script language. To avoid issues such as circular dependencies, there are some basic rules you must follow when writing scripts, or when working in the GUI. Below, we discuss these interfaces and then discuss the basic rules and best practices for working in each interface.

GUI Overview

When you start a GMAT session, the GMAT desktop is displayed and the default mission is loaded. The GMAT desktop has a native look and feel on each platform and most desktop components are supported on all platforms. The components of the desktop are discussed in detail in the Windows GUI section below and the differences for Mac and Linux platforms are discussed in separate sections.

Windows GUI

When you open GMAT on Windows and click Run in the Toolbar, GMAT executes the default mission as shown in the figure below. The tools listed below the figure are available in the GMAT desktop.

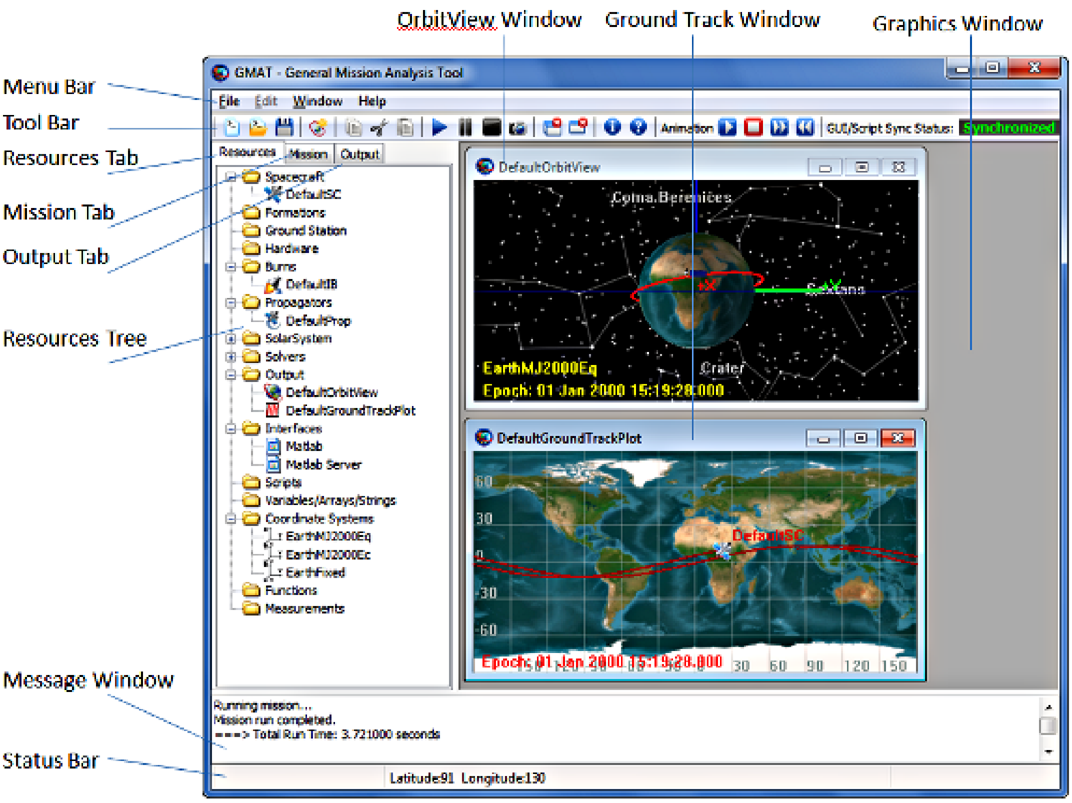


Figure 1. GMAT Desktop (Windows)

Menu Bar

The menu bar contains File, Edit, Window and Help functionality.

On Windows, the **File** menu contains standard **Open**, **Save**, **Save As**, and **Exit** functionality as well as **Open Recent** and **New Mission**. The **Edit** menu contains functionality for script editing and is only active when the script editor is active. The **Window** menu contains tools for organizing graphics windows and the script editor within the GMAT desktop. Examples include the ability to **Tile** windows, **Cascade** windows and **Close** windows. The Help menu contains links to **Online Help**, **Tutorials**, **Forums**, and the **Report An Issue** option links to GMAT's defect reporting system, the **Welcome Page**, and a **Provide Feedback** link.

On Mac, menus are nearly the same, with a few differences: the **File** menu does not contain an **Exit** option - instead, the **Quit GMAT** menu option is on the GMAT menu, as discussed before; tiling and cascading windows are not supported, so those options do not appear under the **Window** menu; currently, email is not supported, so **Provide Feedback** is nonfunctional under the **Help** menu.

Toolbar

The toolbar provides easy access to frequently used controls such as file controls, **Run**, **Pause**, and **Stop** for mission execution, and controls for graphics animation. On Windows and Linux, the toolbar is located at the top of the GMAT window; on Mac, it is located on the left of the GMAT frame. Because the toolbar is vertical, some toolbar options are abbreviated.

GMAT allows you to simultaneously edit the raw script file representation of your mission and the GUI representation of your mission. It is possible to make inconsistent changes in these mission representations. The **GUI/Script Sync Status** indicator located in the toolbar shows you the state of the two mission representations. See the the section called “GUI/Script Interactions and Synchronization” section for further discussion.

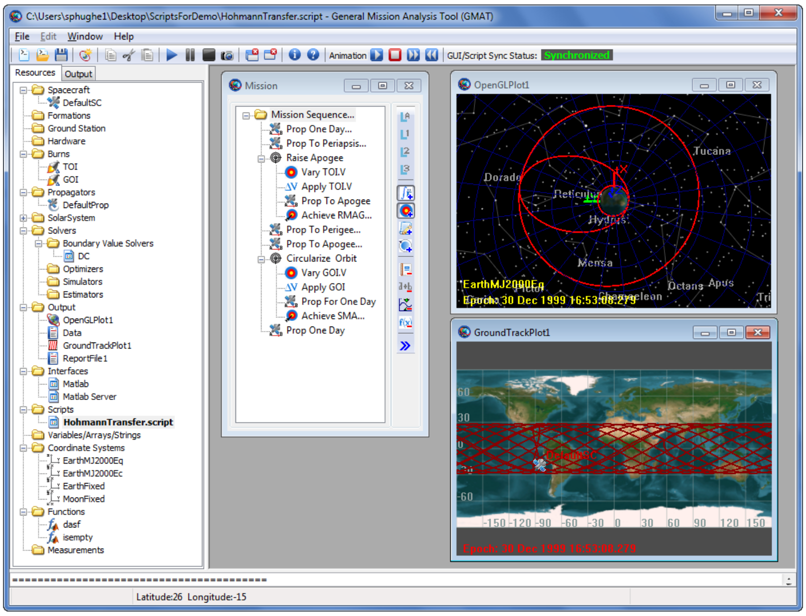
Resources Tab
: The **Resources** tab brings the Resources Tree to the foreground of the desktop.

Resources Tree
: The Resources Tree is a tree structure that displays all configured GMAT resources and organizes them into logical groups. All objects created in a GMAT script using a **Create** command are found in the **Resources Tree** in the GMAT desktop.

Mission Tab
: The **Mission** tab brings the Mission Tree to the foreground of the desktop.

Mission Tree
: The Mission Tree is a tree structure that displays GMAT commands that control the time-ordered sequence of events in a mission. The Mission Tree contains all script lines that occur after the `BeginMissionSequence` command in a GMAT script. You can undock the Mission Tree as shown in the figure below by right-clicking on the **Mission** tab and dragging it into the graphics window. You can also follow these steps:

1. Click on the **Mission** tab to bring the Mission Tree to the foreground.
2. Right-click on the **Mission Sequence** folder in the Mission Tree and select **Undock Mission Tree** in the menu.



Output Tab
: The **Output** tab brings the Output Tree to the foreground of the desktop.

Output Tree
: The **Output Tree** is a tree structure that contains GMAT output such as report files, event reports, and ephemeris files.

Message Window
: When you run a mission in GMAT, information including warnings, errors, and progress are written to the message window. For example, If there is a syntax error in a script file, a detailed error message is written to the message window detailing the error.

Script Interface Overview

The GMAT script editor is a textual interface that supports most functionality in GMAT. In Figure 2 below, the script editor is shown maximized in the GMAT desktop and the items relevant to script editing are labeled.

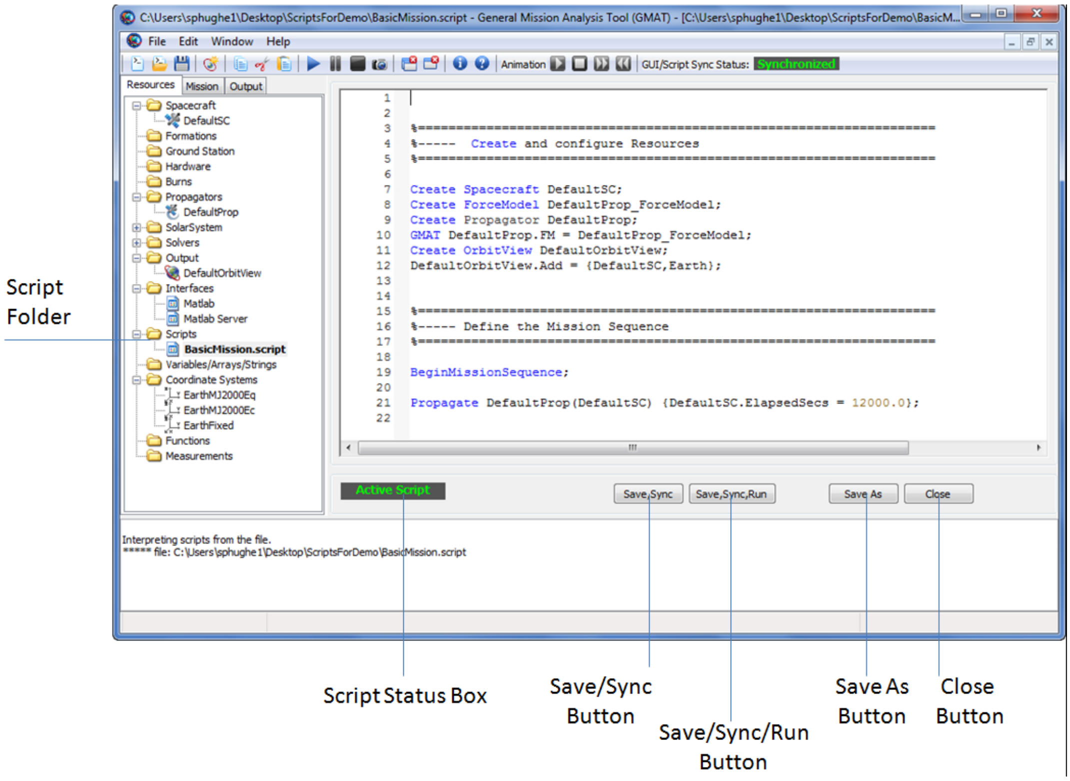


Figure 2. GMAT Script Editor

Script Folder	The GMAT desktop allows you to have multiple script files open simultaneously. Open script files are displayed in the **Scripts** Folder in the Resources Tree. Double click on a script in the **Scripts** folder to open it the script editor. The GMAT desktop displays each script that is open for view in a separate script editor. GMAT uses bold face font to identify which, if any, of the scripts in the **Scripts** folder are loaded into the GUI. Only one script can be loaded into the GUI at a time.
Script Status Box	The **Script Status** box indicates whether or not the script being edited is loaded in the GUI. The box says "Active Script" for the script currently loaded into the GUI and "Inactive Script" for all others.
Save,Sync Button	The **Save,Sync** button saves any script file changes to disk, makes the script active, and synchronizes the GUI with the script.
Save,Sync,Run Button	The **Save,Sync,Run** button saves any script file changes to disk, makes the script active, synchronizes the GUI with the script, and executes the script.
Save As Button	When you click **Save As**, GMAT displays the **Choose A File** dialog box and allows you to save the script using a new file name. After saving, GMAT loads the script into the GUI, making the new file the active script.

Close	The **Close** button closes the script editor.

GUI/Script Interface Interactions and Rules

The GMAT desktop supports a script interface and a GUI interface and these interfaces are designed to be consistent with each other. You can think of the script and GUI as different "views" of the same data: the resources and the Mission Command Sequence. GMAT allows you to switch between views (script and GUI) and have the same view open in an editable state simultaneously. Below we describe the behavior, interactions, and rules of the script and GUI interfaces so you can avoid confusion and potential loss of data.

GUI/Script Interactions and Synchronization

GMAT allows you to simultaneously edit both the script file representation and the GUI representation of your mission. It is possible to make inconsistent changes in these representations. The **GUI/Script Sync Status** window located in the toolbar shows you the state of the two representations, as described in the following table. On Mac, the status is indicated in abbreviated form in the left-hand toolbar. **Synchronized** (green) indicates that the script and GUI contain the same information. **GUI Modified** (yellow) indicates that there are changes in the GUI that have not been saved to the script. **Script Modified** (yellow) indicates that there are changes in the script that have not been loaded into the GUI. **Unsynchronized** (red) indicates that there are changes in both the script and the GUI.

> **Caution**
>
> GMAT will NOT attempt to merge or resolve simultaneous changes in the Script and GUI and you must choose which representation to save if you have made changes in both interfaces.

The Save button in the toolbar saves the GUI representation over the script. The Save/Sync button on the script editor saves the script representation and loads it into the GUI.

How the GUI Maps to a Script

Clicking the **Save** button in the toolbar saves the GUI representation to the script file; this is the same file you edit when working in the script editor. GUI items that appear in the Resources Tree appear before the `BeginMissionSequence` command in a script file and are written in a predefined order. GUI items that appear in the Mission Tree appear after the `BeginMissionSequence` command in a script file in the same order as they appear in the GUI.

> **Caution**
>
> If you have a script file that has custom formatting such as spacing and data organization, you should work exclusively in the script. If you load your script into the GUI, then click **Save** in the toolbar, you will lose the formatting of your script. (You will NOT, however, lose the data.)

How the Script Maps to the GUI

Clicking the **Save/Sync** button on the script editor saves the script representation and loads it into the GUI. When you work in a GMAT script, you work in the raw file that GMAT reads and writes. Each script file must contain a command called `BeginMissionSequence`. Script lines that appear before the `BeginMissionSequence` command create and configure models and this data will appear in the Resources Tree in the GUI. Script lines that appear after the `BeginMissionSequence` command define your mission sequence and appear in the Mission Tree in the GUI. Here is a brief script example to illustrate:

```
Create Spacecraft Sat
Sat.X = 3000
BeginMissionSequence
Sat.X = 1000
```

The line `Sat.X = 3000` sets the x-component of the Cartesian state to 3000; this value will appear on the **Orbit** tab of the **Spacecraft** dialog box. However, because the line `Sat.X = 1000` appears after the `BeginMissionSequence` command, the line `Sat.X = 1000` will appear as an assignment command in the Mission Tree in the GUI.

Basic Script Syntax Rules

- Each script file must contain one and only one `BeginMissionSequence` command.
- GMAT commands are not allowed before the `BeginMissionSequence` command.
- You cannot use inline math statements (equations) before the `BeginMissionSequence` command in a script file. (GMAT considers in-line math statements to be an assignment command. You cannot use equations in the Resources Tree, so you can also not use equations before the `BeginMissionSequence` command.)
- In the GUI, you can only use in-line math statements in an `Assignment` command. So, you cannot type **`3000 + 4000`** or **`Sat.Y - 8`** in the text box for setting a spacecraft's dry mass.
- GMAT's script language is case-sensitive.

Data and Configuration

Below we discuss the files and data distributed with GMAT and that are required for GMAT execution. GMAT uses many types of data files, including planetary ephemeris files, Earth orientation data, leap second files, and gravity coefficient files. This section describes how those files are organized and describe the controls provided so that you can customize the data files GMAT uses at run time.

File Structure

The default directory structure for GMAT is broken into eight main subdirectories, as shown in Figure 3. These directories organize the files and data used to run GMAT, including binary libraries, data files, texture maps, and 3D models. The only two files in the GMAT root directory are `license.txt`, which contains the text of the NASA Open Source Agreement, and `README.txt`, which contains user information for the current GMAT release. A summary of the contents of each subdirectory is described in further detail in the sections below.

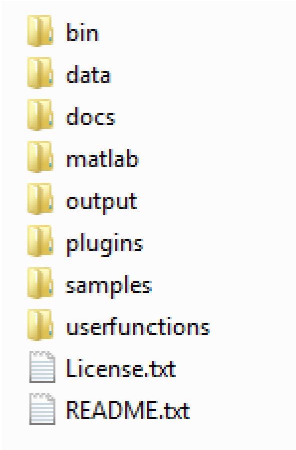


Figure 3. GMAT Root Directory Structure

`bin`

The `bin` directory contains all binary files required for the core functionality of GMAT. These libraries include the executable file (`GMAT.exe` on Windows, `GMAT.app` on Mac, and `GMAT` on Linux) and platform-specific support libraries. The `bin` directory also contains two text files: `gmat_startup_file.txt` and `gmat.ini`. The startup file is discussed in detail in a separate section below. The `gmat.ini` file is used to configure some GUI panels, set paths to external web links, and define GUI tooltip messages.

`data`

The `data` directory contains all required data files to run GMAT and is organized according to data type, as shown in Figure 4 and described below.

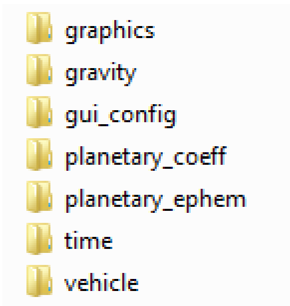


Figure 4. GMAT Data Directory Structure

The `graphics` subdirectory contains data files for GMAT's visualization utilities, as well as application icons and images. The `splash` directory contains the GMAT splash screen that is displayed briefly while GMAT is initializing. The `stars` directory contains a star catalogue used for displaying stars in 3D graphics. The texture folder contains texture maps used for the `OrbitView` 3D graphics resource. The `icons` directory contains graphics files for icons and images loaded at run time, such as the GMAT logo and toolbar icons.

The `gravity` subdirectory contains gravity coefficient files for each body with a default non-spherical gravity model. Within each subdirectory, the coefficient files are named according to the model they represent, and use the extension `.cof`.

The `gui_config` subdirectory contains files for configuring some of the GUI dialog boxes for GMAT resources and commands. These files allow you to easily create a GUI panel for a user-provided plugin, and are also used by some of the built-in GUI panels.

The `planetary_coeff` subdirectory contains the Earth orientation parameters (EOP) provided by the International Earth Rotation Service (IERS) and nutation coefficients for different nutation theories.

The `planetary_ephem` subdirectory contains planetary ephemeris data in both DE and SPK formats. The `de` directory contains the binary digital ephemeris DE405 files for the 8 planets, the Moon, and Pluto developed and distributed by JPL. The `spk` directory contains the DE421 SPICE kernel and kernels for selected comets, asteroids and moons. All ephemeris files distributed with GMAT are in the little-endian format.

The `time` subdirectory contains the JPL leap second kernel `naif0009.tls` and the GMAT leap second file `tai-utc.dat`.

The `vehicle` subdirectory contains ephemeris data and 3D models for selected spacecraft. The `ephem` directory contains SPK ephemeris files, including orbit, attitude, frame, and time kernels. The `models` directory contains 3D model files in 3DS or POV format for use by GMAT's `OrbitView` visualization resource.

docs

The `docs` directory contains end-user documentation, including PDF versions of the Mathematical Specification, Architectural Specification, and Estimation Specification. The GMAT User's Guide is available in the `help` subdirectory in PDF and HTML formats, and as a Windows HTML Help file.

matlab

The `matlab` directory contains M-files required for GMAT's MATLAB interfaces, including the interface to the fmincon optimizer and interfaces for driving GMAT from MATLAB. All files in the `matlab` directory and its subdirectories must be included in your MATLAB path for the MATLAB interfaces to function properly.

output

The `output` directory is the default location for file output such as ephemeris files and report files. If no path information is provided for reports or ephemeris files created during a GMAT session, then those files will be written to the output folder.

plugins

The `plugins` directory contains optional plugins that are not required for use of GMAT. The `proprietary` subdirectory is used for for third-party libraries that cannot be distributed freely and is an empty folder in the open source distribution.

samples

The `samples` directory contains over 30 sample missions, ranging from a Hohmann transfer to libration point station-keeping to Mars B-plane targeting. These files are intended to demonstrate GMAT's capabilities and to provide you with a potential starting point for building common mission types for your application and flight regime. Samples with specific requirements are located in subdirectories such as `NeedMatlab` and `NeedVF13ad`.

`userfunctions`

The `userfunctions` directory contains GMAT and MATLAB functions that are included in the GMAT distribution. You can also store your own custom GMAT and MATLAB functions in these folders.

Configuring GMAT Data Files

You can configure the data files GMAT loads at run time by editing the `gmat_startup_file.txt` file located in the `bin` directory. The startup file contains path information for data files such as ephemeris, earth orientation parameters and graphics files. By editing the startup file, you can customize which files are loaded and used during a GMAT session. Below we describe the customization features available in the startup file. The order of lines in the startup file does not matter.

Leap Second and EOP files

GMAT reads several files that are used for high fidelity modelling of time and coordinate systems: the leap second files and the Earth orientation parameters (EOP) provided by the IERS. The EOP file is updated daily by the IERS. To update your local file with the latest data, simply replace the file `eopc04_08.62-now` in the `data/planetary_coeff` directory. Updated versions of this file are available from the IERS [http://data.iers.org/products/213/14444/orig/eopc04_08.62-now].

There are two leap second files provided with GMAT in the `data/time` directory. The `naif0009.tls` file is used by the JPL SPICE libraries when computing ephemerides. When a new leap second is added, you can replace this file with the new file from NAIF [ftp://naif.jpl.nasa.gov/pub/naif/generic_kernels/lsk/]. GMAT reads the `tai-utc.dat` file for all time computations requiring leap seconds that are not performed by the SPICE utilities. You can modify this file if a new leap second is added by simply duplicating the last row and updating it with the correct information for the new leap second. For example, if a new leapsecond were added on 01 Jul 2013, then you would add the following line to the bottom of `tai-utc.dat` file:

```
2013 JUL 1 =JD 2456474.5 TAI-UTC= 35.0 S + (MJD - 41317.) X 0.0
```

Loading Custom Plugins

Custom plugins are loaded by adding a line to the startup file (`bin/gmat_startup_file.txt`) specifying the name and location of the plugin file. In order for a plugin to work with GMAT, the plugin library must be placed in the folder referenced in the startup file. You specify the path to a plugin file using the "PLUGIN" keyword and specify the file by providing its name without the file extension (.dll on Windows). For example, to load a Windows plugin named `libVF13Optimizer.dll` located in the `plugins/proprietary` directory, you would add this line to your startup file:

```
PLUGIN = ../plugins/proprietary/libVF13Optimizer
```

User-defined Function Paths

If you create custom GMAT or MATLAB functions, you can provide the path to those files and GMAT will locate them at run time. The default startup file is configured so you can place

GMAT function files (with a `.gmf` extension) in the `userfunctions/gmat` directory and place MATLAB functions (with a `.m` extension) in the `userfunctions/matlab` directory. GMAT automatically searches those locations at run time. You can change the location of the search path to your GMAT or MATLAB functions by changing these lines in your startup file to reflect the location of your files with respect to the GMAT `bin` folder:

```
GMAT_FUNCTION_PATH = ../userfunctions/gmat
MATLAB_FUNCTION_PATH = ../userfunctions/matlab
```

If you wish to organize your custom functions in multiple folders, you can add multiple search paths to the startup file. For example,

```
GMAT_FUNCTION_PATH = ../MyFunctions/utils
GMAT_FUNCTION_PATH = ../MyFunctions/StateConversion
GMAT_FUNCTION_PATH = ../MyFunctions/TimeConversion
```

Configuring the MATLAB Interface

GMAT features a MATLAB interface that allows you to run MATLAB functions from within GMAT.

This interface is packaged as an optional GMAT plugin. To use it, make sure the following line is present in your `gmat_startup_file.txt` and has no comment symbol (`#`) in front of it.

```
PLUGIN = ../plugins/libMatlabInterface
```

The MATLAB interface must be able to find your MATLAB installation. The procedure for setting this information varies by platform.

Windows

On Windows, MATLAB must be properly configured in two places: the system `Path` variable and the Windows registry. Both locations must be configured for the same MATLAB version.

1. The following three directories must exist in your system's `Path` variable, where *`<MATLAB>`* is the path to the MATLAB root directory:

   ```
   <MATLAB>/bin/win32
   <MATLAB>/bin
   ```

   If you have multiple versions of MATLAB installed, GMAT will use the one that appears first in the system path.

   > **Caution**
   >
   > The above folders are added to your system path during MATLAB installation. However, for some versions of MATLAB (e.g. 2010a), MATLAB and Windows are distributed with libraries that have the same name. This may cause the Windows libraries to load instead of the MATLAB libraries. As a result, you may need to put the folders above at the beginning of your system path.

2. When you install MATLAB, it automatically registers itself as a COM server in the Windows registry. If you have mulple versions of MATLAB installed, it may be necessary to re-register a certain version manually. This can be done by running the following command. This may require administrator privileges.

```
matlab.exe -regserver
```

3. Add GMAT's MATLAB files to your MATLAB path. This can be done by placing the following line in a file named `startup.m` in your user MATLAB directory, where *<GMAT>* is the path to your GMAT root directory.

```
addpath(genpath('<GMAT>/matlab'));
```

Mac OS X

On Mac OS X, to use MATLAB with GMAT, you must set the `MATLABFORGMAT` environment variable in your `environment.plist` file, located in the `.MacOSX` directory in your home folder. This environment variable should point to the location of your MATLAB installation (application bundle). GMAT will not interface with MATLAB without this environment variable being set.

The current Mac application includes the ability to make calls to MATLAB functions from within GMAT, but does not support calls MATLAB to GMAT (including the `fmincon` optimizer).

Note that when GMAT opens MATLAB, it will open X11 first (as is required for MATLAB execution). GMAT currently does not automatically close X11 after quitting MATLAB, so you will need to quit X11 manually.

To add the environment variable:

1. If the `environment.plist` file already exists in your `.MacOSX` directory, edit the file using the Property List Editor to add the `MATLABFORGMAT` variable and set it to point to the location of your MATLAB application (e.g. `/Applications/MATLAB_R2010a/MATLAB_R2010a.app`).
2. If you do not have an `environment.plist` in your `.MacOSX` directory, using a terminal window:
    1. Create the `.MacOSX` directory as a subdirectory in your home folder (if it does not exist).
    2. Open the Property List Editor, create the `MATLABFORGMAT` variable as described above.
    3. Save the property list as `environment.plist` in your `.MacOSX` directory.

You must logout and log back in for this to take effect.

Other Resources

If you have further questions or need help for GMAT, or want to provide feedback, here are some additional resources:

- Official Homepage: `http://gmat.gsfc.nasa.gov`
- User Forum: `http://gmat.ed-pages.com/forum`
- Wiki: `http://gmat.ed-pages.com/wiki`
- Mailing Lists and Project Resources: `http://sourceforge.net/projects/gmat`
- Blog: `http://gmat.sourceforge.net/blog`
- Documentation: `http://gmat.sourceforge.net/docs`
- Bug Tracker: `http://pows003.gsfc.nasa.gov/bugzilla`
- Official Contact: `<gmat@gsfc.nasa.gov>`

Part II. Creating Your First Mission

Table of Contents

Simulating an Orbit

Objective and Overview

> **Note**
>
> The most fundamental capability of GMAT is to propagate spacecraft, which said another way is to simulate the orbital motion. The ability to propagate spacecraft is used in nearly every practical aspect of space mission analysis from simple orbital predictions--when will the International Space Station be over my house?--to complex analyses that determine the thirster firing sequence required to send a spacecraft to the Moon or Mars.

This tutorial will teach you how to use GMAT to propagate a spacecraft. You will learn how to configure a **Spacecraft** and a **Propagator**, and then set up a **Propagate** command to propagate the spacecraft to orbit perigee, which is the point of minimum distance between a spacecraft and Earth. The basic steps in this tutorial are:

1. Configure the**Spacecraft** and define its epoch and orbital elements.
2. Configure the **Propagator**.
3. Modify the default **OrbitView** to visualize the spacecraft trajectory.
4. Modify the **Propagate** command to propagate the spacecraft to the perigee.
5. Run the mission and analyze the results.

Configure the Spacecraft

In this section, you will rename a **Spacecraft** and set the **Spacecraft**'s initial epoch and classical orbital elements. You'll need GMAT open, with the Default Mission loaded. To load the Default Mission click **New Mission** in the **Toolbar** or start a new GMAT session.

Rename the Spacecraft

1. In the **Resources Tree**, right-click **DefaultSC**, and select **Rename**.
2. In the **Rename** dialog box, type **Sat** .
3. Click **OK**.

Set the Spacecraft Epoch

1. In the **Resources Tree**, double-click on **Sat**. Click the **Orbit** tab if it is not already selected.
2. In the **Epoch Format** box, select **UTCGregorian**. You'll see the value in the **Epoch** field change to the UTC Gregorian epoch format.
3. In in the **Epoch** field, type **22 Jul 2014 11:29:10.811**.
4. Click **Apply** to save these changes.

Set the Keplerian Orbital Elements

1. In the **StateType** box select **Keplerian** . In the **Elements** list, you will see the GUI reconfigure to display the Keplerian state representation.
2. In the **SMA** box, type **83474.318**.

3. Set the remaining orbital elements as shown in the table below.

Table 1. Sat Orbit State Settings

Field	Value
ECC	0.89652
INC	12.4606
RAAN	292.8362
AOP	218.9805
TA	180

4. Click **OK**.
5. In the **Toolbar**, Click the **Save** button. If this is the first time you have saved the mission, you'll be prompted to provide a name and location for the file.

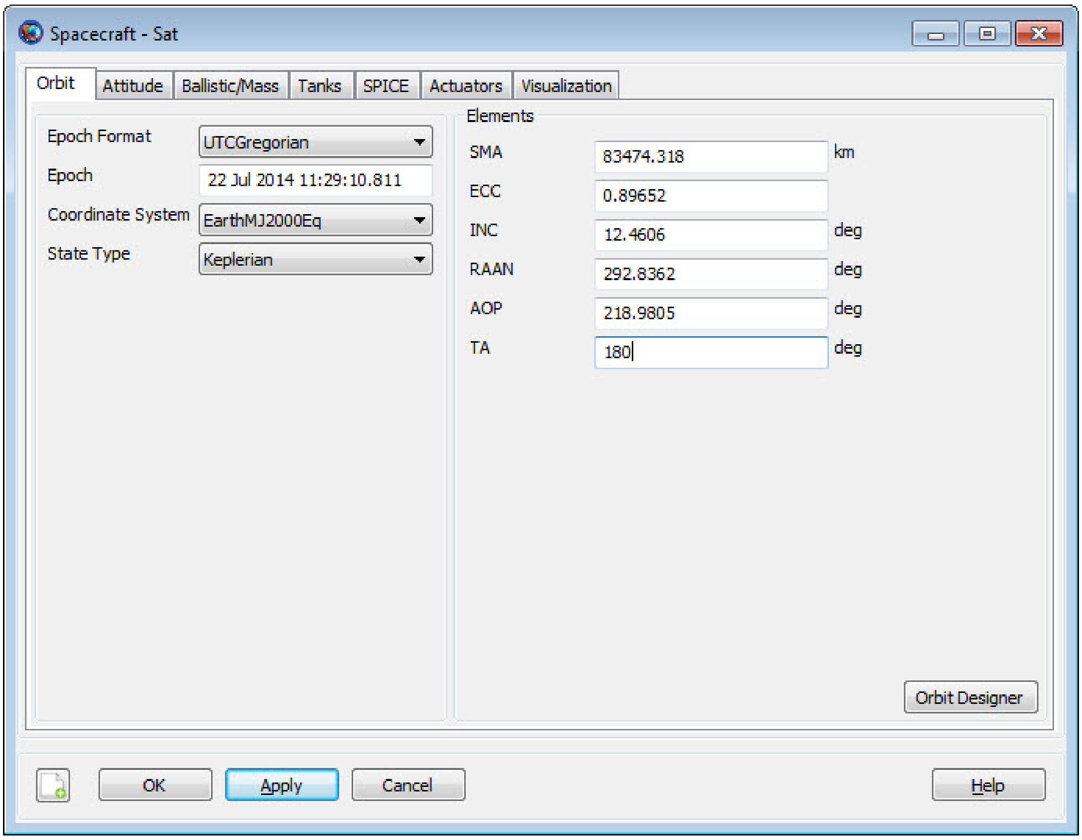


Figure 5. Spacecraft State Setup

Configure the Propagator

In this section you'll rename the **Propagator**, and configure the force model.

Rename the Propagator

Here you'll rename the propagator with a more descriptive name:

1. In the **Resources Tree**, double-click **DefaultProp**, and select **Rename**.

2. In the **Rename** dialog box, type **LowEarthProp**.
3. Click **OK**.

Configure the Force Model

Now configure the force model. For this tutorial you will use an Earth 10x10 spherical harmonic model, Jacchia-Roberts atmospheric model, solar radiation pressure, and point mass perturbations from the Sun and Moon.

1. In the **Resources Tree**, double-click **LowEarthProp**,
2. In the **Gravity** list, type **10** in the **Degree** box.
3. In the **Order** box, Type **10**.
4. In **Atmosphere Model** box, select **JacchiaRoberts**.
5. Click the **Select** button next to the **Point Masses** text box. This opens the **CelestialBodySelect** dialog box.
6. In the **Available Bodies** list, click **Sun**, then click **->** to add **Sun** to the **Selected Bodies** list.
7. Add the moon (named Luna in GMAT) using the same procedure above.
8. Click **Use Solar Radiation Pressure** to toggle it on.
9. Click **OK**.

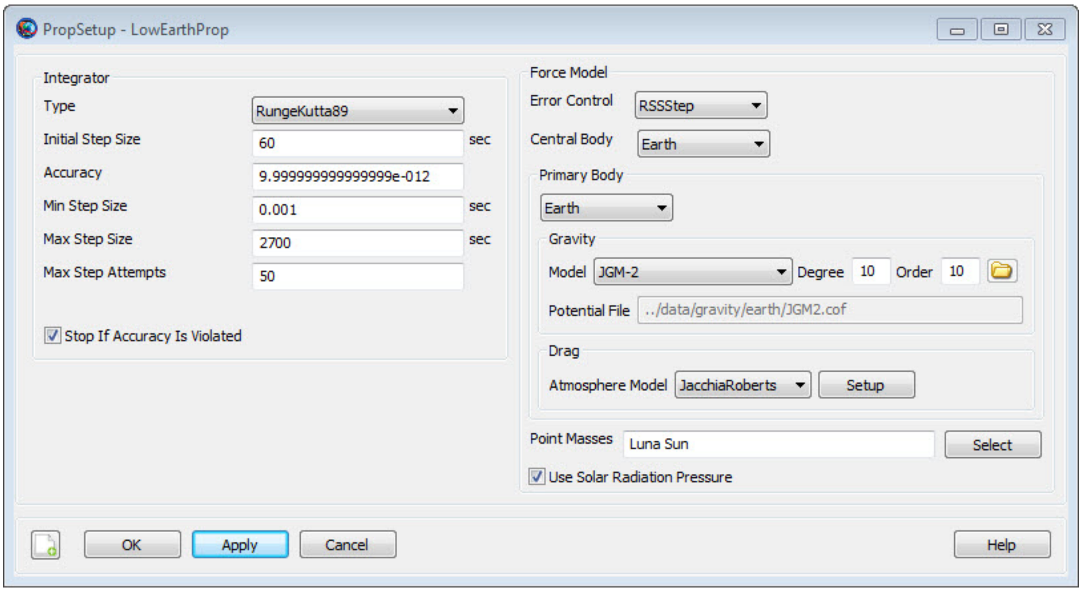


Figure 6. Force Model Configuration

Configuring the Orbit View Plot

Below you will configure an **OrbitView** plot so you can visualize **Sat** and its trajectory. The orbit of **Sat** is highly eccentric. To view the entire orbit in the plot, we need to adjust the settings of **DefaultOrbitView**.

1. In the **Resources Tree**, double-click on **DefaultOrbitView**.
2. In the three boxes next to **ViewPointVector**, type the values **-60000**, **30000**, and **20000** respectively.
3. In the **Drawing Options** list, un-check **DrawXY Plane**.
4. Click **OK**.

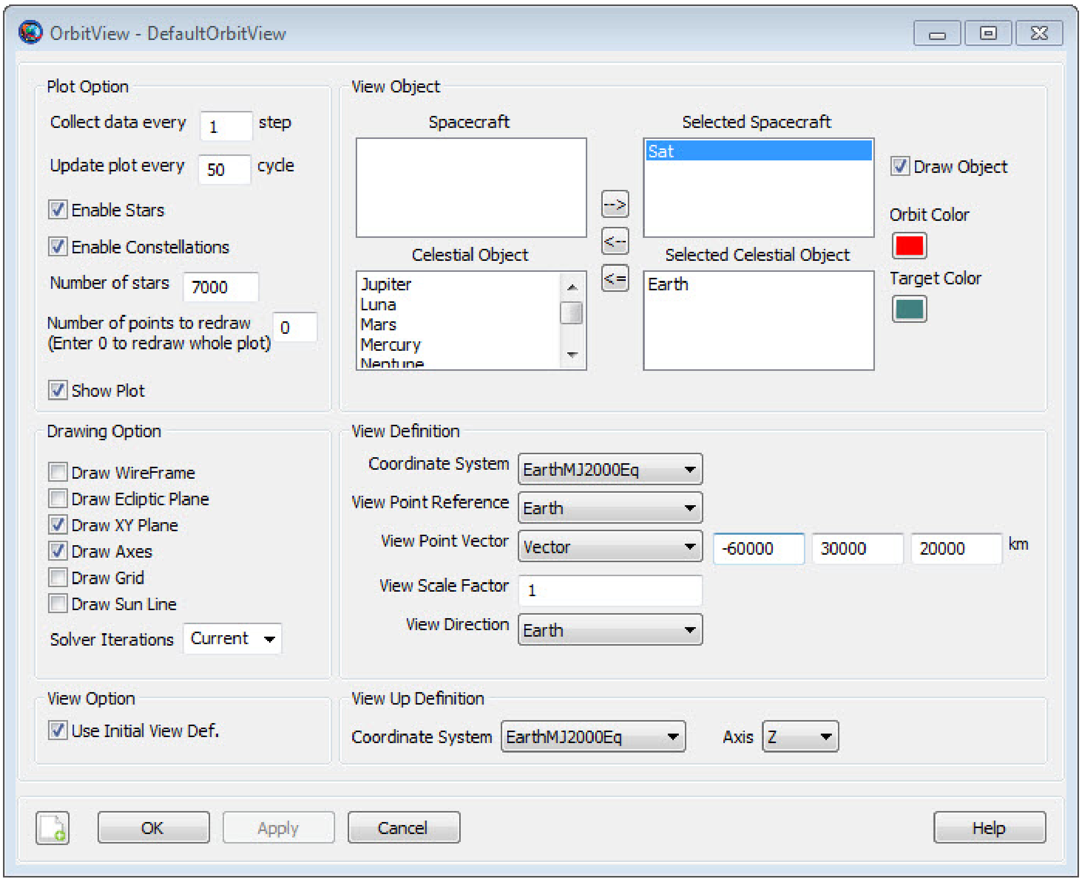


Figure 7. DefaultOrbitView Configuration

Configure the Propagate Command

This is the last step in the tutorial before running the mission. Below you will configure a Propagate command to propagate (model the motion of) Sat to orbit perigee.

1. Click the **Mission** tab to bring the **Mission Tree** to the foreground.
2. Double-click on **Propagate1**.
3. In the **Stopping Conditions** list, right click the (**...**) button next to Sat.ElapsedSecs to bring up the **Parameter Select Dialog**
4. In the **Object** list, select **Sat** if it is not already selected. This directs GMAT to associate the stopping condition with the spacecraft **Sat**.
5. In the **Object Properties** list, double-click **Periapsis** to add it to the **Selected Values** list as shown in Figure 8.

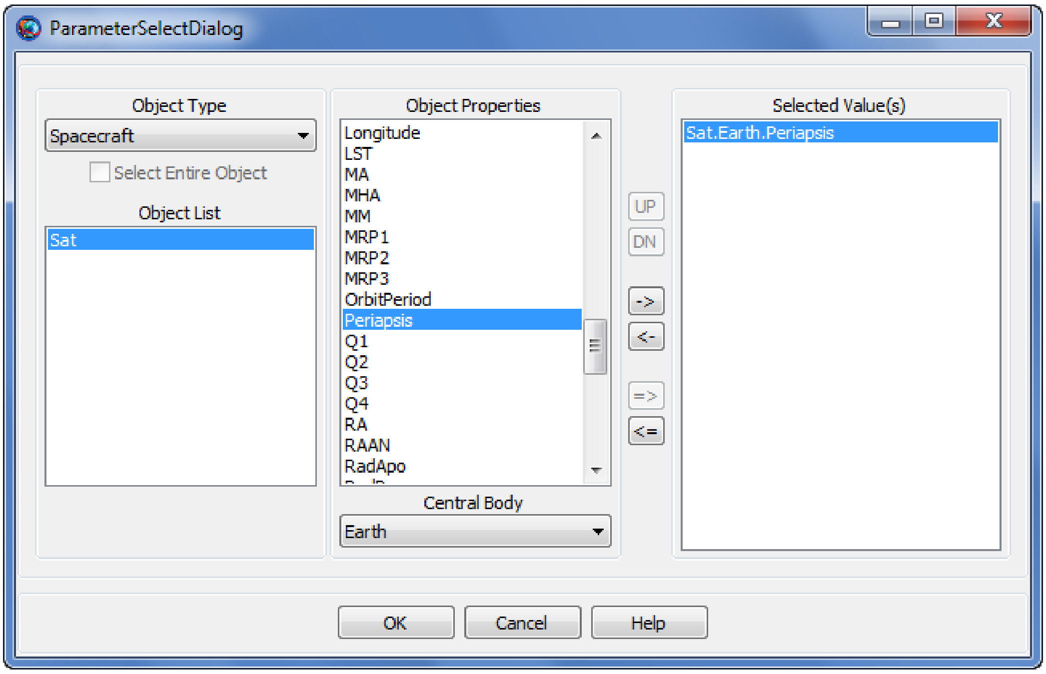


Figure 8. Propagate Command Parameter Select Dialog Configuration

6. Click **OK**.

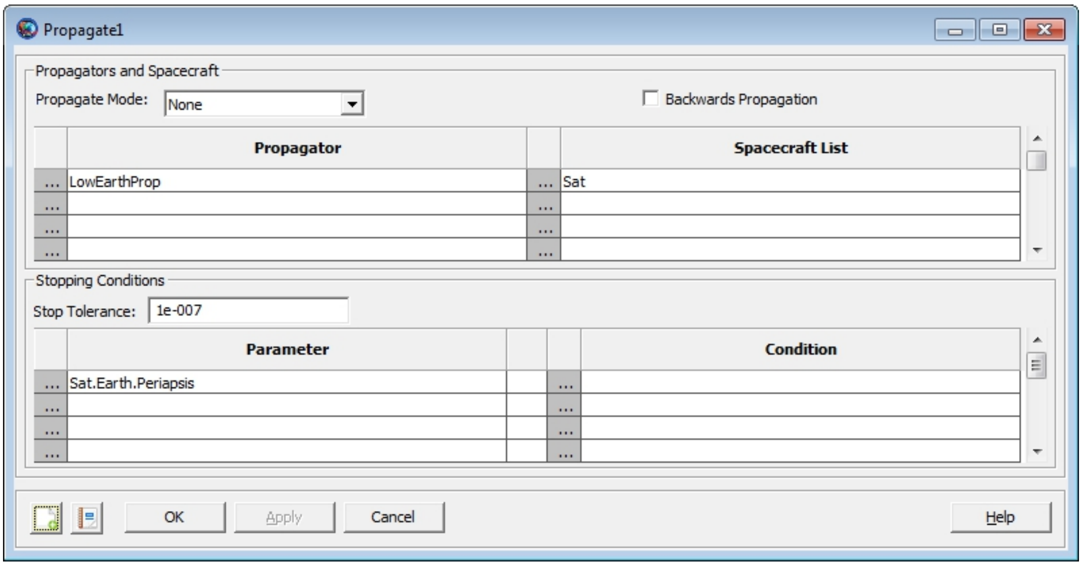


Figure 9. Propagate Command Configuration

Run and Analyze the Results

Congratulations, you have now configured your first GMAT mission and are ready to run the mission and analyze the results.

1. Click the **Save** button in the **Toolbar** to save your mission.
2. Click the **Run** button in the **Toolbar**.

You will see GMAT propagate the orbit and stop at orbit periapsis. Figure 10 below illustrates what you should see after correctly completing this tutorial. Here are a few things you can try to explore the results of this tutorial:

1. Manipulate the **Orbit View** plot using your mouse to orient the trajectory so that you can to verify that at the final location the spacecraft is at perigee.
2. Click the **Mission Tab** to bring the **Mission Tree** to the foreground.
3. Left-click on **Propagate1** and select **Command Summary** to see data on the final state of **Sat**.
4. What values for longitude and latitude do you see for **Sat**.
5. Are the values for the final longitude and latitude of **Sat** consistent with the **Ground Track** plot **Sat**?
6. Close the **Command Summary** dialog box.
7. Click in the **DefaultOrbitView** graphics window to bring the window to the foreground.
8. Click **Start Animation** in the **Toolbar** to animate the mission and watch the orbit propagate from apogee to perigee.

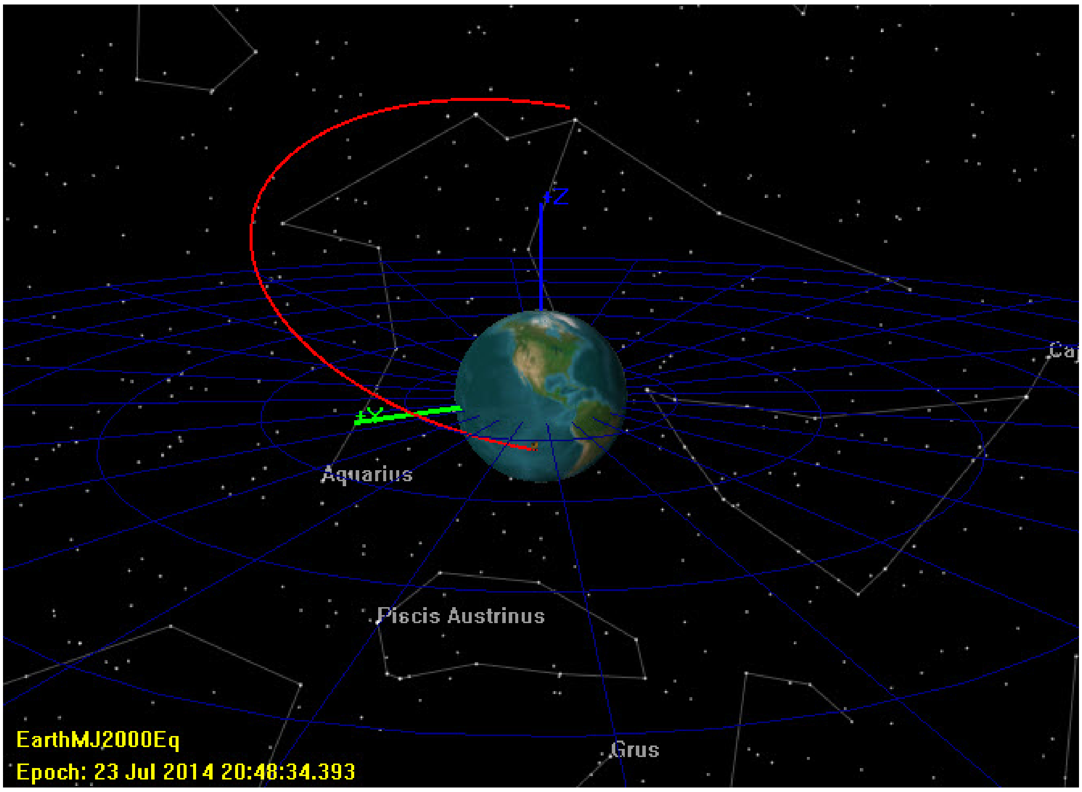


Figure 10. Orbit View Plot after Mission Run

Part III. How-tos

Table of Contents

Configuring a Spacecraft

Setting the Initial Epoch

You can configure the initial epoch of a Spacecraft using several time systems (TAI, TDB, UTC, etc) and in several formats (Gregorian Date, Modified Julian Date). In this How To you'll learn how to set a spacecraft's epoch in UTC Gregorian format. Starting from the default mission:

1. In the **Resources** tree, double-click on **DefaultSC** to open it.
2. Make sure the **Orbit** tab is selected
3. In the **EpochFormat** list, select **UTCGregorian**.
4. Type **`04 July 2014 09:30:15.235`** in the **Epoch** text box.

The script for the epoch settings configured above is shown below.

```
Create Spacecraft DefaultSC;
GMAT DefaultSC.DateFormat = UTCGregorian;
GMAT DefaultSC.Epoch = '04 Jul 2014 09:30:15.235';
```

Configuring the Orbit

To learn how to to define the initial state for a spacecraft orbit, you'll configure GMAT to propagate the International Space Station (ISS). Starting from the default mission, first set the initial epoch:

1. In the **Resources** tree, right-click on **DefaultSC**, and click **Rename**.
2. Type **ISS** in the **Rename** Dialog Box and then click **OK**.
3. On the **Resources** tree, double-click on **ISS**.
4. Make sure the **Orbit** tab is selected.
5. Click on the **Epoch Format** drop-down menu and select **UTCGregorian**.
6. Type **`21 Oct 2011 14:01:29.130`** in the **Epoch** text box.

Now follow the steps below to set the orbital state for ISS:

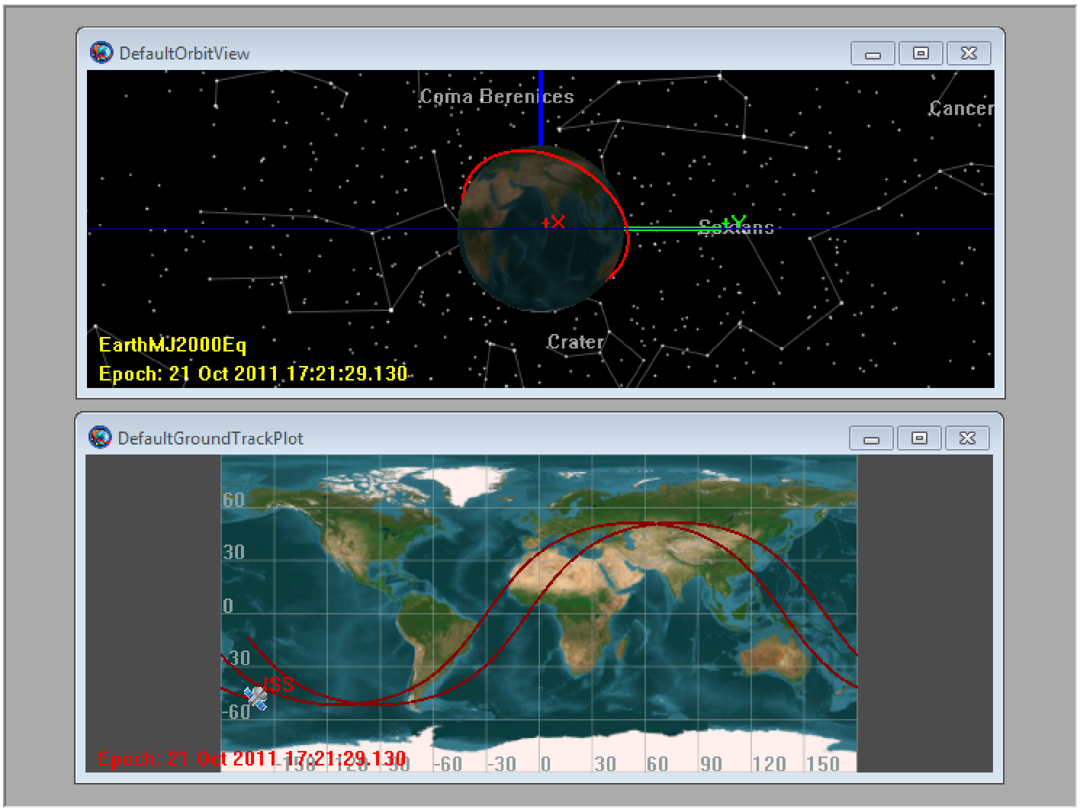
1. In the **Resources** tree, double-click on **DefaultSC** to open it.
2. Make sure the **Orbit** tab is selected.
3. Click the **State Type** drop-down menu and select **Keplerian**.
4. In the **Elements** parameter list on the **Spacecraft** dialog box, type **`6771.907`** in the **SMA** text box.
5. Type **`0.00103`** in the **ECC** text box.
6. Type **`51.597`** in the **INC** text box.
7. Type **`244.300`** in the **RAAN** text box.
8. Type **`353.735`** in the **AOP** text box.
9. Type **`199.683`** in the **TA** text box.
10. Click **OK**.

The script for the spacecraft state configured above is show below.

```
Create Spacecraft ISS
ISS.DateFormat=UTCGregorian
ISS.Epoch = 21 Oct 2011 14:01:29.130
ISS.SMA = 6771.907
ISS.ECC = 0.00103
```

```
ISS.INC = 51.597
ISS.RAAN = 244.300
ISS.AOP = 353.735
ISS.TA = 199.683
```

Click **Run** in the Toolbar and you will see plots like those shown below in the **Graphics Window**.



Configuring Physical Properties

To configure the physical properties of a spacecraft, start from the default mission and perform the following steps.

1. In the **Resources** tree, double-click on **DefaultSC** to open it.
2. Click on the **Ballistic/Mass** tab in the **Spacecraft** dialog box.
3. Type **`450`** in the **DryMass** text box.
4. Type **`2.0`** in the **Coefficient of Drag** text box.
5. Type **`1.7`** in the **Coefficient of Reflectivity** text box.
6. Type **`10.5`** in the **Drag Area** text box.
7. Type **`12.5`** in the **SRP Area** text box.

The script for the physical settings configured above is shown below.

```
Create Spacecraft DefaultSC
DefaultSC.DryMass = 450
DefaultSC.Cd = 2.0
DefaultSC.Cr = 1.7
DefaultSC.DragArea = 10.5
DefaultSC.SRPArea = 12.5
```

Propagating a Spacecraft

Configuring the Force Model

In the example below, you'll learn how to configure a high fidelity propagator for low Earth orbits. Starting from the default mission:

1. In the **Resources** tree, double-click on **DefaultProp** to open it.
2. In the **Gravity** section, type 21 in the **Degree** box.
3. Type 21 in the **Order** box.
4. Click on the **Atmosphere Model** drop-down list and select MSISE90 .
5. Click the **Select** button next to the **PointMasses** box.
6. Click on **Sun** and then click the right arrow **->** to add the **Sun** to your force model.
7. Add **Moon**, and **Jupiter** using the same steps as above.
8. Click **OK** on the **CelestialBodySelect** dialog box.
9. Click on the **UseSolarRadiationPressure** check box. The box should be checked now.
10. Click **OK** on the **PropSetup** dialog box.

The script for the force model configured above is shown below.

```
Create ForceModel DefaultProp_ForceModel;
DefaultProp_ForceModel.CentralBody = Earth;
DefaultProp_ForceModel.PrimaryBodies = {Earth};
DefaultProp_ForceModel.PointMasses = {Jupiter, Luna};
DefaultProp_ForceModel.Drag = MSISE90;
DefaultProp_ForceModel.SRP = On;
DefaultProp_ForceModel.ErrorControl = RSSStep;
DefaultProp_ForceModel.GravityField.Earth.Degree = 21;
DefaultProp_ForceModel.GravityField.Earth.Order = 21;
DefaultProp_ForceModel.GravityField.Earth.PotentialFile = 'JGM2.cof';
```

Propagating for a Duration

GMAT can propagate a spacecraft for a duration of time, such as 60 seconds, 30 days, or one orbit period. Starting from the default mission:

1. Click the **Mission** tab to show the **Mission** tree.
2. Double-click **Propagate1**. The default mission is configured to propagate the DefaultSC spacecraft for 12000 seconds.
3. In the **Parameter** column, to the left of **DefaultSC.ElapsedSecs**, click .... This will display a window allowing you to choose a new type of duration parameter.
4. In the **Object Properties** list, click **ElapsedDays**, then click -> to add it to the **Selected Value(s)** list.
5. Click OK.
6. In the **Condition** column, double-click the value **`0.0`** and enter **`30`** instead.
7. Click OK, then click Run.

GMAT will propagate the spacecraft for 30 days.

Propagating to an Orbit Condition

GMAT can propagate a spacecraft to a specific orbit condition, such as periapsis, an altitude value, or a latitude value. Starting from the default mission:

1. Click the **Mission** tab to show the **Mission** tree.
2. Double-click **Propagate1**. The default mission is configured to propagate the DefaultSC spacecraft for 12000 seconds.
3. In the **Parameter** column, to the left of **DefaultSC.ElapsedSecs**, click .... This will display a window allowing you to choose a new type of duration parameter.
4. In the **Object Properties** list, click **Periapsis**.
5. In the **Central Body** list, make sure **Earth** is selected. Then click -> to add it to the **Selected Value(s)** list.
6. Click OK to close the **ParameterSelectDialog** window.
7. Click OK, then click Run.

GMAT will propagate the spacecraft until it reaches periapsis.

Reporting Data

GMAT provides several ways to report mission data (such as altitude or delta-V values) to plain text files. GMAT can report data at each integration time step in the mission or at specific mission events, such as periapsis passage. The report functionality is controlled via the **ReportFile** resource and the **Report** and **Toggle** commands.

Reporting Data During a Propagation Span

You can report data at each integration step in the mission sequence by creating a **ReportFile** resource and adding data to it. Starting from the default mission:

1. On the **Resources** tree, right-click the **Output** folder, point to **Add**, and click **ReportFile**.
2. Double-click the **ReportFile1** resource.
3. In the **Parameter List** area, click **Edit**.
4. In the **Selected Value(s)** list, click **DefaultSC.EarthMJ2000.X** and click **<-** to remove it from the list.
5. In the **Object Properties** list, click **Altitude** and click **->** to add it to the **Selected Value(s)** list.
6. Add **DefaultSC.A1ModJulian** to the **Selected Value(s)** list if it doesn't already exist.
7. Click **OK**, then in the **ReportFile - ReportFile1** dialog box, click **OK** again.
8. Click **Run**. To view the generated report, on the **Output** tree, double-click **ReportFile1**.

The script for the report data configured above is shown below.

```
Create ReportFile ReportFile1;
GMAT ReportFile1.Add = {DefaultSC.A1ModJulian, DefaultSC.Earth.Altitude};
```

Reporting Data at a Specific Mission Event

You can report data to a **ReportFile** at a desired location in the mission sequence using the **Report** command. In this How To, you'll learn how to report spacecraft altitude at orbit apogee. Starting from the default mission:

1. In the **Resources** tree, right-click on the **Output** folder, point to **Add**, and click **ReportFile**.
2. In the **Output** folder, double-click on **ReportFile1** to open it.
3. In the **Parameter List** area, click the **Edit** button.
4. In the **Selected Values** list, click on **DefaultSC.EarthMJ2000Eq.X** and click the left arrow **<-** to remove it from the list.
5. Remove **DefaultSC.A1ModJulian** from the **Selected Value(s)** list using the step above.
6. Click **OK** to close the **ParameterSelectDialog** box and then click **OK** again to close the **ReportFile1** dialog box.

Now you will configure the **Propagate1** command to propagate to periapsis then issue a command to report to **ReportFile1**.

1. Click on the **Mission** tab to bring the mission tree to the foreground.
2. In the **Mission** tree, double-click on **Propagate1** to open it.
3. In the **Stopping Conditions** list, click on the ellipses **...** to the left of **DefaultSC.ElapsedSecs**.
4. In the **Object Properties** list, click on **Apoapsis** then click on the right arrow **->** to add it to the **Selected Value(s)** list.

5. Click **OK** to close the **ParameterSelectDialog** box and then click **OK** again to close the **Propagate1** dialog box.
6. Right-click on **Propagate1**, point to **Insert After**, and then select **Report**.
7. Double-click **Report1** and click on the **View** button.
8. Click on the remove all button **<=** to remove all items from the **Selected Value(s)** list.
9. Click on **TA** in the **Object Properties** list then click the right arrow **->** to add it to the **Selected Value(s)** list.
10. Add **Altitude** to the **Selected Value(s)** list using the same step as above.
11. Click **OK** to close the **ParameterSelectDialog** box and then click **OK** again to close the **Report1** dialog box.
12. In the Toolbar, click **Run**.
13. Click the **Output** tab.
14. In the **Reports** folder, click **ReportFile1** to you will see the requested data.

The script for the report data configured above is shown below.

```
Create ReportFile ReportFile1;
BeginMissionSequence;
Propagate DefaultProp(DefaultSC) {DefaultSC.Earth.Apoapsis};
Report ReportFile1 DefaultSC.Earth.TA DefaultSC.Earth.Altitude;
```

Creating a CCSDS Ephemeris File

The CCSDS Orbit Ephemeris Message (OEM) is a standardized text-based ephemeris format. In GMAT, you can easily create an OEM file with your desired interpolation order and data frequency. Starting from the default mission:

1. In the **Resources** tree, right-click the **Output** folder, point to Add, and click EphemerisFile. A new resource called **EphemerisFile1** appears in the tree.
2. Double-click **EphemerisFile1** to open it.
3. Make sure that in the **File Format** list, **CCSDS-OEM** is selected.
4. Click **Ok**.
5. Click **Run**. The OEM file will be written to a file named `EphemerisFile1.eph` in GMAT's output folder.

Creating an SPK Ephemeris File

An SPK ephemeris is a binary file format used by the SPICE Toolkit created by NAIF. GMAT can write spacecraft state information to this format using your desired interpolation order and data frequency. Starting from the default mission:

1. In the **Resources** tree, right-click the **Output** folder, point to Add, and click EphemerisFile. A new resource called **EphemerisFile1** appears in the tree.
2. Double-click **EphemerisFile1** to open it.
3. In the **File Format** list, click **SPK**.
4. In the **File Name** box, replace the default value with **`EphemerisFile1.bsp`**. An SPK ephemeris requires the `.bsp` extension.
5. Click **Ok**.
6. Click **Run**. The SPK file will be written to a file named `EphemerisFile1.bsp` in GMAT's output folder.

Visualizing Data

Manipulating the 3D Orbit View

GMAT's OrbitView resource offers a three-dimensional realistic view of your mission trajectory in any coordinate system or viewpoint you choose. The view itself can be manipulated using the mouse. Starting from the default mission:

1. Click Run. This will run the mission and will result in a **DefaultOrbitView** window on the GMAT desktop. The default view is centered at the Earth, in an Earth-centered inertial reference frame.
2. With the left mouse button, drag in the **DefaultOrbitView** window. This will rotate the view about the center of the active coordinate system (in this case, the center of the Earth).
3. With the right mouse button, drag left-to-right. This will zoom the view out from the center of the active coordinate system. Dragging right-to-left will zoom the view in.
4. With the wheel button (or middle button), drag up and down. This will rotate the view about an axis perpendicular to the screen.

Configuring the Ground Track Plot

GMAT's ground track plot can display one or more spacecraft on a two-dimensional map of a celestial body. You can choose which spacecraft are displayed, and which celestial body to use. Keeping the Earth as the central body, let's add a second spacecraft to the default plot. Starting with the default mission, first add a new spacecraft:

1. Right-click the **Spacecraft** folder and click Add Spacecraft to add **Spacecraft1**.
2. In the **Mission** tree, double-click **Propagate1**.
3. Under **Spacecraft List**, click ... to the left of **DefaultSC**.
4. In the **Available SpaceObject** list, click **Spacecraft1** and click -> to move it to the **SpaceObject Selected** list. Then click OK. This adds **Spacecraft1** to the **Spacecraft List** for **Propagate1**.
5. Click Apply, then click OK.

Then add the new spacecraft to the ground track plot:

1. In the **Resources** tree, in the **Output** folder, double-click **DefaultGroundTrackPlot**.
2. In the **Selected Objects** list, select **Spacecraft1**.
3. Click OK, then click Run.

After the run is complete, the **DefaultGroundTrackPlot** window will show the trajectory of the default spacecraft and **Spacecraft1** on a map of Earth.

Creating a 2D Plot

GMAT offers an XYPlot resource that allows you to visualize the relationship between multiple parameters (for example, orbit altitude and time). Starting from the default mission:

1. In the **Resources** tree, right-click the **Output** folder, point to Add, and click XYPlot.
2. Double-click the new **XYPlot1** resource. The default y-axis parameter is the Cartesian "X" position of the spacecraft.
3. Click Edit Y to change the y-axis parameter.
4. In the **Selected Value(s)** list, click **DefaultSC.EarthMJ2000Eq.X** and click <- to remove it from the list.

5. In the **Object Properties** list, click **Altitude**.
6. In the **Central Body** list, make sure **Earth** is selected, then click -> to add it to the **Selected Value(s)** list.
7. Click OK in the **ParameterSelectDialog** window, then click OK again in the **XYPlot - XYPlot1** window.
8. Click Run.

A new plot window will appear.

Part IV. Tutorials

Table of Contents

Simple Orbit Transfer

Objective and Overview

Final result: HohmannTransferDesign.script [scripts/HohmannTransferDesign.script]

> **Note**
>
> One of the most common problems in space mission design is to transfer from one circular orbit to another circular,coplanar orbit. Circular, coplanar transfers are used to raise low-Earth orbits that have degraded due to the effects of atmospheric drag. They are also used to transfer from a low-Earth orbit to a Geosynchronous orbit and to send spacecraft to Mars. There is a well known sequence of maneuvers, called the Hohmann transfer, that performs a circular, coplanar transfer using the least possible amount of fuel. A Hohmann transfer employs two maneuvers. The first maneuver raises the orbital apogee (or lowers orbital perigee) to the desired altitude and places the spacecraft in an elliptical transfer orbit. At the apogee (or perigee) of the elliptical transfer orbit, a second maneuver is applied to circularize the orbit

In this tutorial, you will use GMAT to perform a Hohmann transfer from a low-Earth parking orbit to a Geosynchronous mission orbit. This requires a **Target** sequence to determine the required maneuver magnitudes to achieve the desired final orbit conditions. In order to focus on the configuration of the **Target** sequence, you will make extensive use of the default configurations for spacecraft, propagators, and maneuvers. The **Target** sequence employs two velocity-direction maneuvers and two propagation sequences. The purpose of the first maneuver is to raise orbit apogee to 42165 km. The purpose of the second maneuver is to nearly circularize the orbit and yield a final eccentricity of 0.005. The basic steps of this tutorial are:

1. Create and configure a **Differential Corrector**.
2. Modify the **DefaultOrbitView** to visualize the trajectory.
3. Create two default **ImpulsiveBurns**.
4. Create a **Target** sequence to (1) raise apogee to GEO altitude and (2) circularize the orbit.
5. Run the mission and analyze the results.

Configure Maneuver, Differential Corrector, and Graphics

For this tutorial, you'll need GMAT open, with the Default Mission loaded. To load the Default Mission click **New Mission** in the **Toolbar** or start a new GMAT session. We will use the default configurations for a spacecraft (**DefaultSC**), a propagator (**DefaultProp**), and maneuvers. **DefaultSC** is configured to a near circular orbit and **DefaultProp** is configured to use Earth as the central body with a gravity model of degree and order 4. The default impulsive burn model uses the Velocity Normal Bi-normal (VNB) coordinate system. You may want to open the dialog boxes for these objects and inspect them more closely as we will leave the settings of those objects at their default values.

Create the Differential Corrector

You will need a **Differential Corrector** later in this tutorial so we'll create one now. You can leave the settings at their defaults.

1. In the **Resource Tree**, locate the **Solvers** folder and expand it if it is minimized.

2. Right-click the **Boundary Value Solvers** folder, point to **Add**, and then select **Differential Corrector**.

Modify the default Orbit View

We need to make minor modifications to DefaultOrbitView so that the entire final orbit will fit in the graphics window.

1. In the **Resource Tree**, double-click the DefaultOrbitView.
2. Set the values shown in the table below.

Table 2. DefaultOrbitView settings

Field	Value
Solver Iterations	**Current**
ViewUpDefintion	**X**
ViewPointVector boxes	**0**,**0**, and **120000** respectively.

3. Click **OK**.

Create the Maneuvers.

We need two **ImpulsiveBurn** objects for this tutorial. Below, you'll rename the default **ImpulsiveBurn** and create a new maneuver.

1. In the **Burns** folder in the **Resource Tree**, right-click on **DefaultIB**, select **Rename**.
2. In the **Rename** dialog box, type **TOI**, an abbreviation for Transfer Orbit Insertion.
3. Right-click on the **Burns** folder, point to **Add**, and select **ImpulsiveBurn**.
4. Rename the new **ImpulsiveBurn** object to **GOI,** an abbreviation for Geosynchronous Orbit Insertion.

Configure the Target Sequence

Now you will configure a **Target** sequence to solve for the maneuver values required to raise the orbit to Geosynchronous altitude and circularize the orbit. We'll begin by creating the **Target** sequence and then discuss the function of each command. Finally, we'll configure the commands for our problem. To allow us to focus on the Target sequence, we'll assume you have learned how to propagate an orbit to a desire condition by taking the Create Your First Mission Tutorial.

Configure the Initial Propagate Sequence

1. Click on the **Mission tab** to bring the **Mission Tree** to the foreground.
2. Configure **Propagate1** to propagate to Periapsis. The procedures are discussed in Creating Your First Mission. You can optionally leave **Propagate1** with default settings.

Create the Target Sequence

Now create the commands necessary to perform the **Target** sequence. Figure 11 illustrates the configuration of the **Mission Tree** after you have complete the steps in this section. We'll discuss the **Target** Sequence after it has been created.

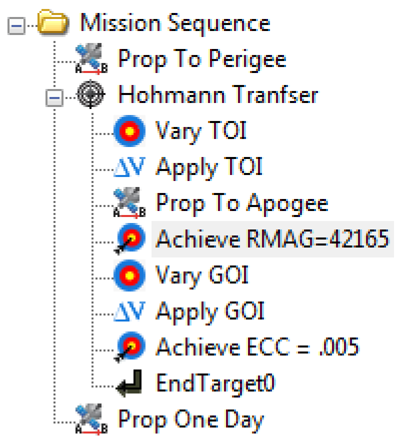


Figure 11. Target Sequence for the Hohmann Transfer

To create the **Target** Sequence:

1. In the **Mission Tree**, right-click on **Prop To Perigee**, point to **Insert After**, and select **Target**.
2. Right-click on **Target1** and select **Rename**.
3. In the **Rename** dialog box, type **Hohmann Transfer**, and click **OK**.
4. Right-click on **EndTarget0**, point to **Insert After**, and select **Propagate**.
5. Right-click on **Hohmann Transfer**, point to **Insert After**, and select **Vary**.
6. Rename **Vary1** to **Vary TOI**.
7. Complete the **Target** sequence by adding the commands inTable 3 after **Vary TOI** in the **Target** sequence.

Table 3. Additional Target Sequence Commands

Command	Name
Maneuver	**Apply TOI**
Propagate	**Prop to Apogee**
Achieve	**Achieve RMAG = 42165**
Vary	**Vary GOI**
Maneuver	**Apply GOI**
Achieve	**Achieve ECC = 0.005**

Note

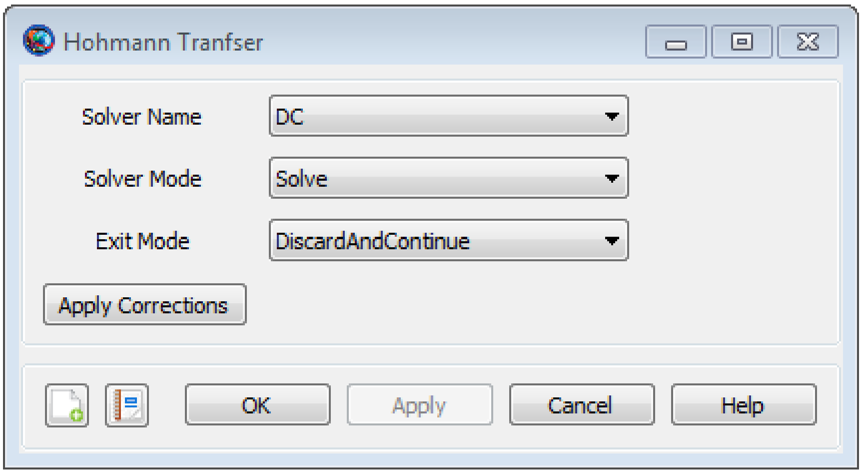
Let's discuss what the **Target** sequence does. We know that two maneuvers are required to perform the Hohmann transfer. We also know that for our current mission, the final orbit radius must be 42165 and the final orbital eccentricity must be 0.005. However, we do NOT know the size (delta v magnitudes) of the maneuvers that precisely achieve the desired orbital conditions. You use the **Target** sequence to solve for those precise maneuver values. But, you must tell GMAT what controls are available, (in this case two maneuver values) and what conditions must be satisfied (in this case orbital radius and eccentricity). You accomplish this using the **Vary** and **Achieve** commands. Using the **Vary** command, you tell GMAT what you would like it to solve for -- in this case the Delta V values for **TOI** and **GOI**. You use the **Achieve** command to tell GMAT what conditions the solution must satisfy--in this case the final orbital conditions.

Configure the Target Sequence

Let's configure the target sequence.

Configure the Hohmann Transfer Command

1. Double-click **Hohmann Transfer** and change **ExitMode** to **SaveAndContinue**.
2. Change **ExitMode** to **SaveAndContinue.** This will save the solution of the targeting problem after you run it later in the tutorial.
3. Click **OK**.



Configure the Vary TOI Command

1. Double-click **Vary TOI**. Notice that the variable in the **Variable in the Setup** list is **TOI.Element1**. Element1 of **TOI** is the velocity component of TOI in the local VNB system. That's what we need, so we'll keep it.
2. In the **InitialValue** box, enter 1.0.
3. In the **MaxStep** box, enter 0.5.
4. Click **OK**.

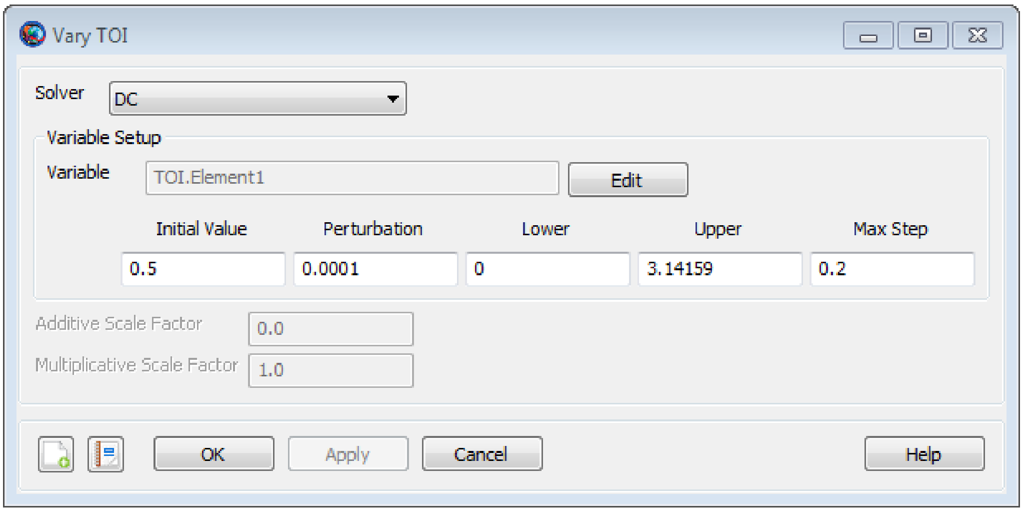
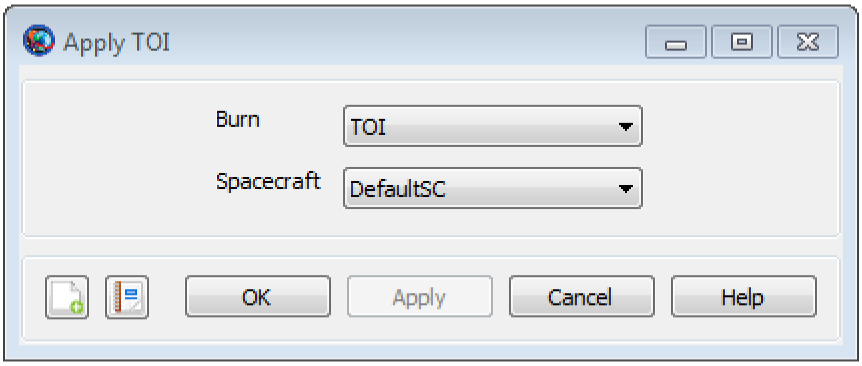


Figure 12. Vary TOI dialog box.

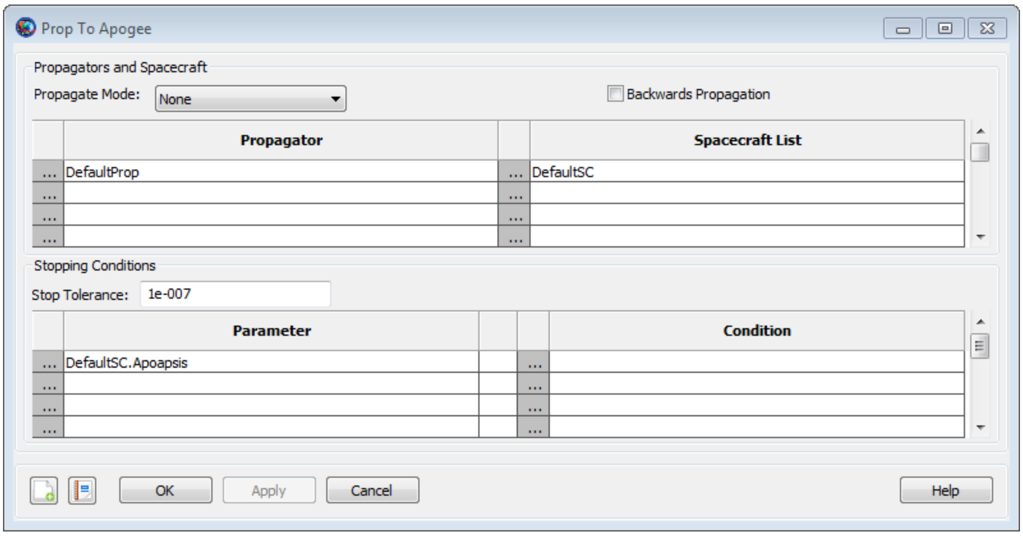
Configure the Apply TOI Command

1. Double-click **Apply TOI** . Notice that the command is already set to apply **TOI** to **DefaultSC**, so we don't need to change anything here.
2. Click OK.



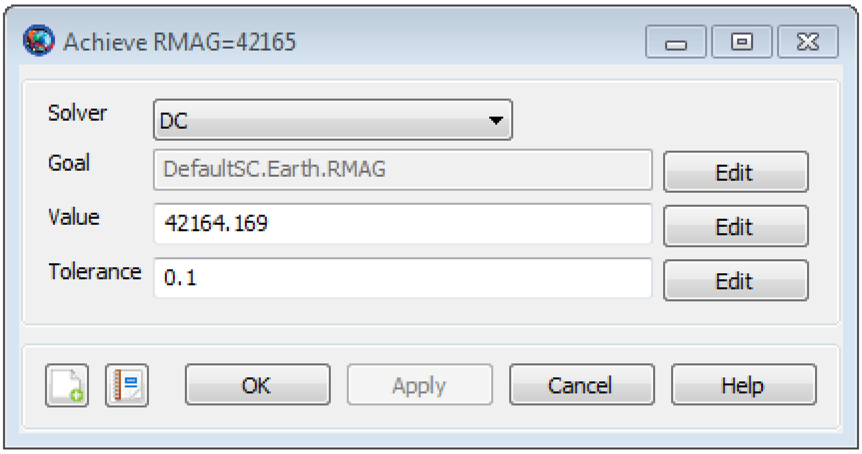
Configure the Prop to Apogee Command

1. Double-click **Prop to Apogee**.
2. In the **Parameter** list, click in the box that says **DefaultSC.ElapsedSecs**.
3. Type **DefaultSC.Apoapsis**.
4. Click **OK**.



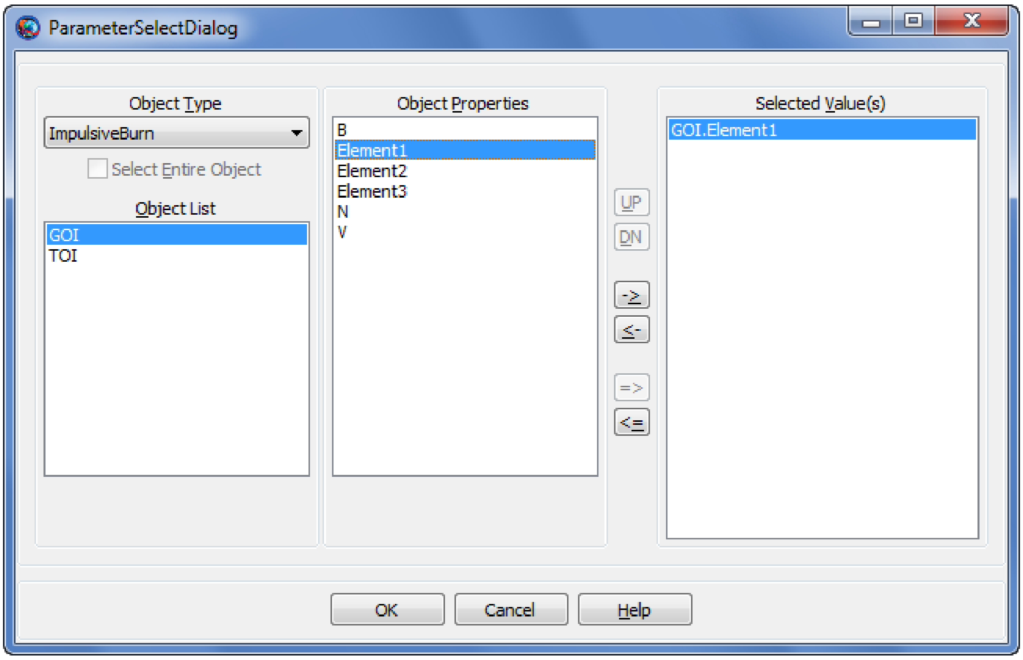
Configure the Achieve RMAG = 42165 Command

1. Double-click **Achieve RMAG = 42165**.
2. Notice that the goal is set to **DefaultSC.Earth.RMAG** - which is what we need, so we make no changes here.
3. In the Value box, enter **42164.169**, a more precise number for the radius of a Geostationary orbit.
4. **Click** OK.

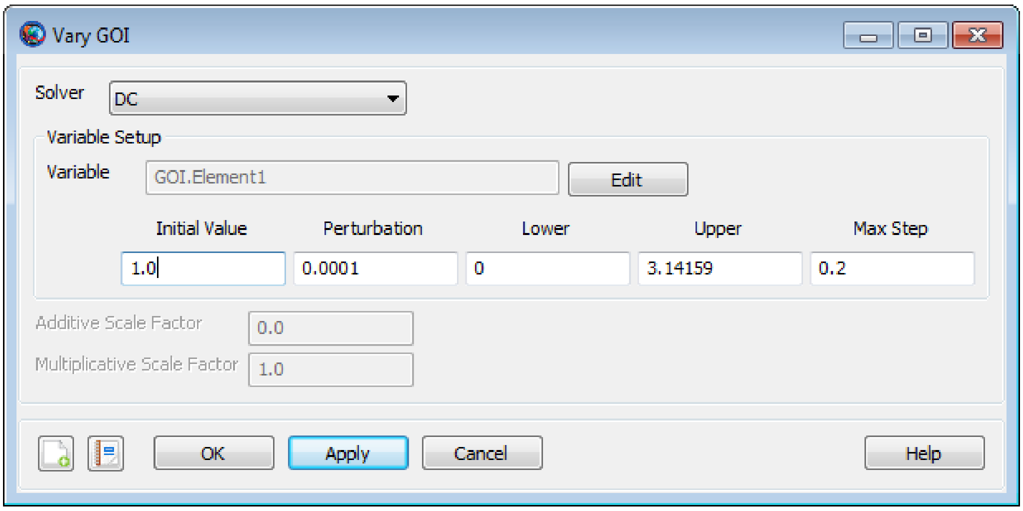


Configure the Vary GOI Command

1. Double-click **Vary GOI Command**.
2. Next to the **Variable** text box, click the **Edit** button.
3. Under **Object List**, click on **GOI**.
4. In the **Object Properties** list, double-click on **Element1**. (See the image below for results.)



5. Click **OK** to close the **Parameter Select** dialog box.
6. In the **Initial Value** box, type **1.0**.
7. In the **MaxStep** text box, type **0.2**.
8. **Click** OK.



Configure the Apply GOI Command

1. Double-click **Apply GOI**.
2. In the **Burn** box, select **GOI**
3. Click **OK**.

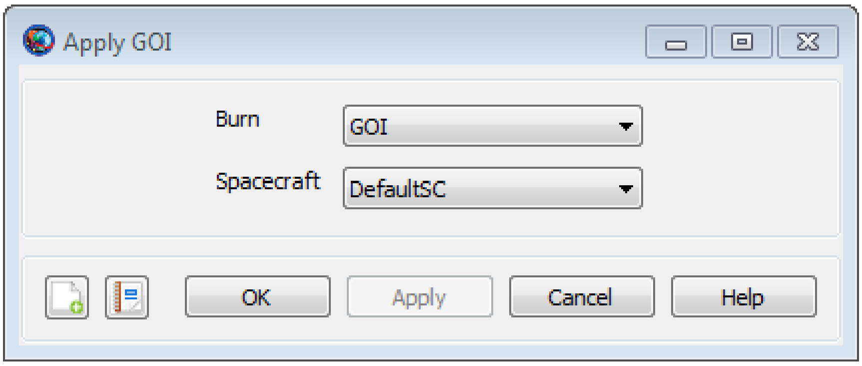


Figure: Maneuver2 Command

Configure the Achieve ECC = 0.005 Command

1. Double-click **Achieve ECC = 0.005**.
2. Next to the **Goal** box, click the **Edit** button.
3. In the **Object Properties** list, double-click **ECC** .
4. Click **OK** to close the **Parameter Select** dialog Box.
5. In the **Value** box, type **0.005**.
6. In the **Tolerance** box, type to **0.0001**.
7. Click OK.

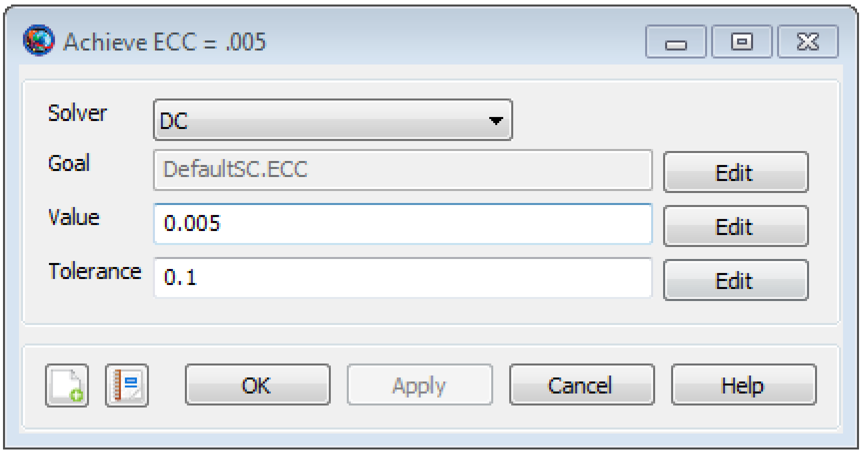
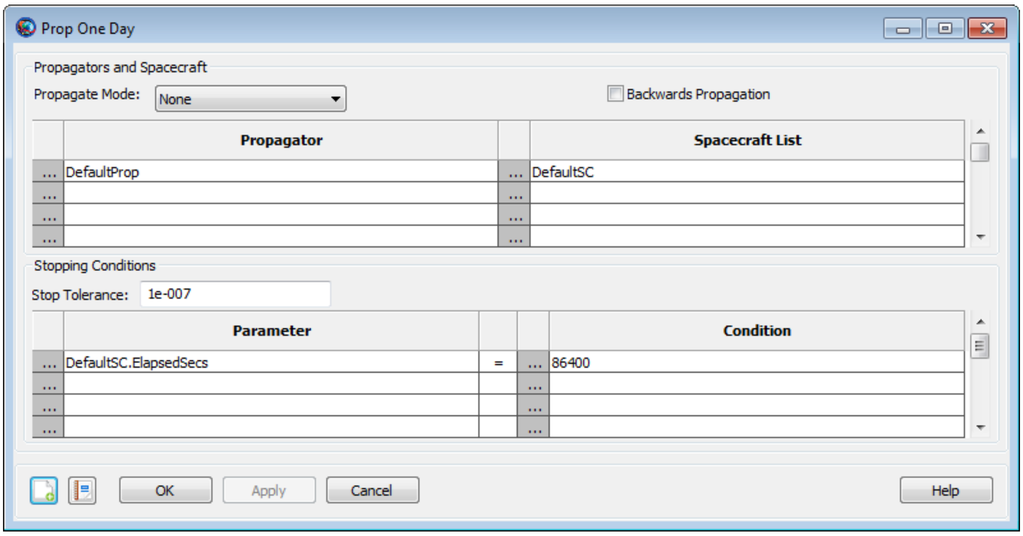


Figure: Achieve2 Command

Configure the Prop One Day Command

A Blank subsection.

1. Double-click **Prop One Day**.
2. In the **Condition** list, click in the box that says **12000**, and then type **86400** .
3. Click **OK**.



Running the Mission

Before running the mission, click **Save** in the **Toolbar** and save your file to the desired location. Now click on the **Run** in the **Toolbar**. As the mission runs, you will see GMAT solve the targeting problem and the iterations and perturbations are shown in light blue in the **DefaultOrbitView** window. After the mission is complete, the **OrbitView** should appear similar to the image shown below.

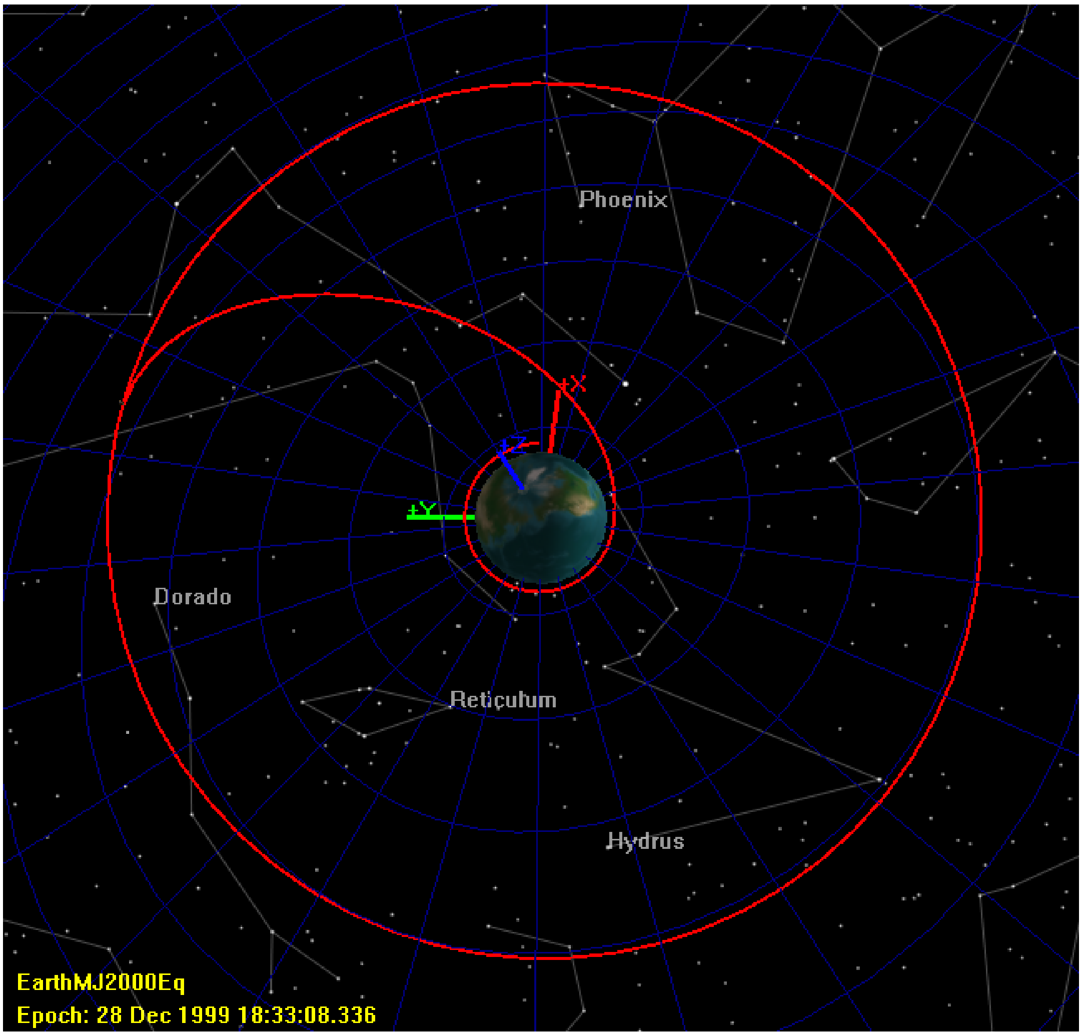


Figure: Output After Final Propagate Sequence

You can save the resulting solution so that if you make small changes to the problem and re-target, the initial guess for subsequent runs will use the solution from your work above.

1. Double-click on **Hohmann Transfer** in the **Mission Tree**.
2. Left-click on **Apply Corrections**.
3. Rerun the mission by clicking the **Run** button in the **Toolbar**. If you inspect the results in the message window, you will see that the **Target** sequence converges in one iteration because you gave provided the solution as the new initial guess.

Part V. Reference Guide

Table of Contents

Resources

Name

Array — A two-dimensional numeric array variable

Synopsis

```
Create Array name[rows,columns];
name(row,column) = value;
…
```

Description

An array is a numeric variable that can contain multiple values in either one or two dimensions (i.e. a matrix).

Fields

$ITEM NAME. WILL MOST LIKELY USE COURIER NEW HTML FORMATTING$	$ITEM DESCRIPTION$	
	Default	$ITEM DEFAULT VALUE$
	Limits	$ITEM OPTIONS$
	Units	$UNITS$
$ITEM NAME. WILL MOST LIKELY USE COURIER NEW HTML FORMATTING$	$ITEM DESCRIPTION$	
	Default	$ITEM DEFAULT VALUE$
	Limits	$ITEM OPTIONS$
	Units	$UNITS$

Interactions

Report Commands	Report commands can be used to retrieve information within arrays or from the entire array.

Examples

Example 1. Creating an array

This example creates an empty one-dimensional array with 5 elements.

```
Create Array Array1[1,5];
```

Example 2. Creating and populating a matrix

This example creates the identity matrix of size 2 and names it `I`:

```
Create Array I[2,2];
I(1,1) = 1;
I(1,2) = 0;
I(2,1) = 0;
I(2,2) = 1;
```

Name

Barycenter — A barycenter.

Synopsis

```
Create Barycenter name
name.BodyNames = {bodyName1,bodyName2,...,bodyNameN}
```

Description

A barycenter is the center of mass of one or more celestial bodies and can be used as the origin of a CoordinateSystem, a reference point in an OrbitView, or as one of the points in a LibrationPoint.

Fields

BodyNames	The BodyNames field is a list that contains the bodies used to define a barycenter. In a script, the list must be surrounded by curly braces. (i.e. BaryCenterName.BodyNames = { Earth, Luna })	
	Default	Earth, Luna
	Limits	Sun, Mercury, Venus, Earth, Luna, Mars, Jupiter, Saturn, Uranus, Neptune, Pluto, or any user-defined celestial body. At least one body must be selected!
	Units	N/A

Interactions

Coordinate Systems	Interacts with the barycenter object selecting it as the origin of the coordinate system or as a primary or secondary point for defining the axes.
OpenGL Plot	It can be selected as a celestial object to be drawn onto the plot, a View Point Reference, or the View Direction within the dialog box.

Examples

```
Create Barycenter EarthMoonBary;
GMAT EarthMoonBary.BodyNames = {Earth, Luna};
```

Name

Coordinate Systems — A coordinate system.

Synopsis

```
Create CoordinateSystem name
name.field = value
```

Description

Coordinate Systems are critical to GMAT for several reasons. They are what every object represented within the software is referenced to. They are used by GMAT as the basis for which all calculations are made. They also provide the reference for any OpenGL Plot that is created.

Interactions

Thruster	The Thruster object allows you to set a coordinate system as its reference.
Spacecraft	In the spacecraft dialog box you may change what coordinate system the spacecraft's position is defined in reference, whatever the state type may be set as.
OpenGL Plot	Coordinate Systems are very key to the display of OpenGL Plots. They rely on coordinate systems to set how exactly the view of the plot will look using both the View Definition and View Up Definition sections of the OpenGL Plot dialog box.
Parameter Select Dialog Box	Whenever you may select a parameter using the parameter select dialog box, you have the option of selecting certain options such X, Y, Z, and several others that will require to set a coordinate system for them to reference.

Examples

```
Create CoordinateSystem EarthMJ2000Eq;
GMAT EarthMJ2000Eq.Origin = Earth;
GMAT EarthMJ2000Eq.Axes = MJ2000Eq;
GMAT EarthMJ2000Eq.UpdateInterval = 60;
GMAT EarthMJ2000Eq.OverrideOriginInterval = false;
```

Name

Differential Corrector — A differential corrector.

Synopsis

```
Create DifferentialCorrector name
name.field = value
```

Description

A differential corrector is a numerical solver for solving two-point boundary value problems. The DC in GMAT uses a simple shooting method where the derivatives are determined using finite differencing. In the mission sequence, you use the differential corrector object in a Target sequence to solve two-point value problems. For example, differential correctors are often used to determine the maneuver components required to achieve desired orbital conditions, say, B-plane conditions at a planetary flyby.

You must create and configure a differential corrector object according to your application by setting numerical properties of the solver such as tolerance and maximum iterations. You can also select among different output options that show increasing levels of information for each differential corrector iteration.

The allowable settings for a differential corrector are shown in the GUI screen shots and reference table below. You can learn more about how to use a DC in a targeting sequence by reading the help files for Target, Vary, and Achieve.

Fields

MaximumIterations	The MaximumIterations field allows the user to set the maximum number of iterations the differential corrector is allowed during the attempt to find a solution. If the maximum iterations is reached, GMAT exits the target loop and continues to the next command in the mission sequence. In this case, the objects retain their states as of the last nominal pass through the targeting loop.	
	Default	25
	Limits	Integer >= 1
	Units	N/A
DerivativeMethod	The DerivativeMethod field allows the user to choose between one-sided and central differencing for numerically determining the Jacobian matrix.	
	Default	ForwardDifference
	Limits	ForwardDifference, BackwardDifference, CentralDifference
	Units	N/A
ShowProgress	When the ShowProgress field is set to true, then data illustrating the progress of the differential correction process are written to the message window. The message window is updated with information on the current control variable values and the contraint variances for both on perturbation and iteration passes. When the ShowProgress field is set to false, no information on the progress of the differential correction process is displayed.	
	Default	true
	Limits	true, false

	Units	N/A
ReportStyle	The ReportStyle field allows the user to control the amount and type of information written to the file defined in the ReportFile field. Currently, the Normal and Concise options contain the same information: the Jacobian, the inverse of the Jacobian, the current values of the control variables, and achieved and desired values of the constraints. Verbose contains values of the perturbation variables in addition to the data for Normal and Concise. Debug contains detailed script snippets at each iteration for objects who have control variables.	
	Default	Normal
	Limits	Normal, Concise, Verbose, Debug
	Units	N/A
ReportFile	The ReportFile field allows the user to specify the path and file name for the differential correction report.	
	Default	DifferentialCorrectorDCName
	Limits	Filename consistent with OS
	Units	N/A

Object and Command Interactions

The Differential Corrector does not interact directly with any resource objects.

The Differential Corrector is used in the following mission sequence commands:

- Target
- Vary
- Achieve

Examples

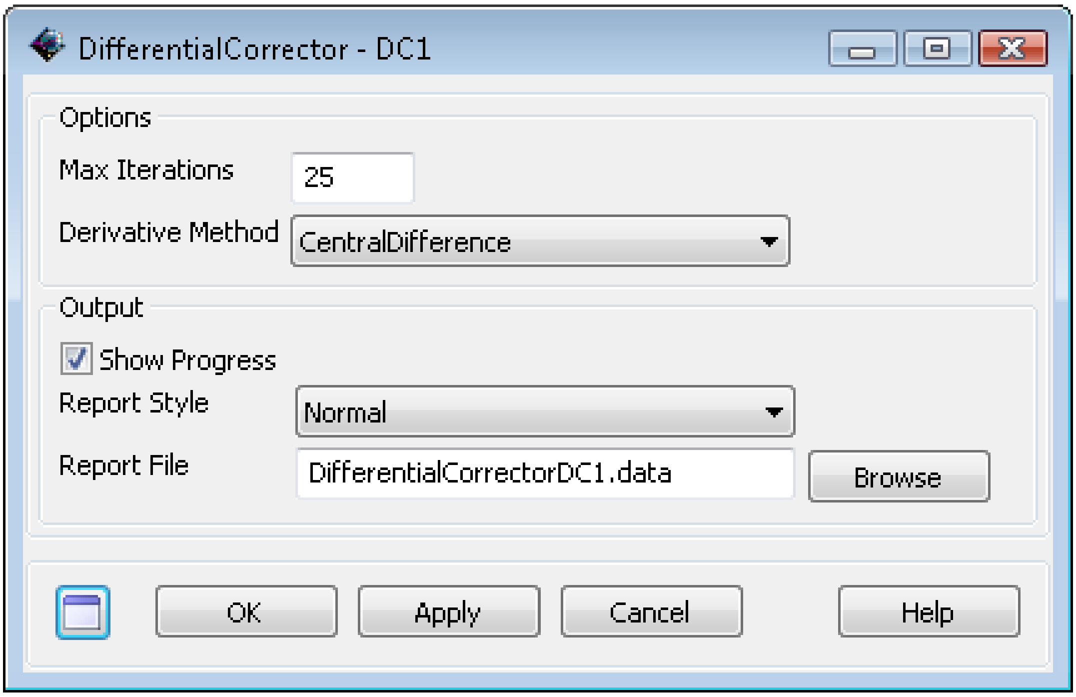


Figure: Default Name and Settings for the Differential Corrector Dialog Box

```
Create DifferentialCorrector DefaultDC;
```

```
DefaultDC.ShowProgress = true;
DefaultDC.ReportStyle = 'Normal';
DefaultDC.TargeterTextFile = 'DifferentialCorrectorDefaultDC.data';
DefaultDC.MaximumIterations = 25;
DefaultDC.UseCentralDifferences = false;
```

Name

EphemerisFile — An ephemeris file.

Synopsis

```
Under Construction.
```

Description

Under Construction.

Fields

Field Name	Description....	
	Type	Fill This In.
	Default	Fill This In.
	Limits	Fill This In.
	Units	Fill This In.

Examples

Example 3. Example Script

```
%
```

Name

EphemerisPropagator — Under Construction.

Synopsis

```
Under Construction.
```

Description

Under Construction.

Fields

Field Name	Description....	
	Type	Fill This In.
	Default	Fill This In.
	Limits	Fill This In.
	Units	Fill This In.

Examples

Example 4. Example Script

```
%
```

Name

Finite Burn — A finite burn.

Synopsis

```
Create FiniteBurn name
name.field = value
```

Description

The Finite Burn object is used when a continuous propulsion is desired. Impulsive burns happen instantaneously and through a Maneuver command, while finite burns occur until the End Finite Burn command is reached in the mission sequence and are typically coupled with Propagate commands.

Fields

Origin	Together the Origin and Axes fields describe the coordinate system in which a maneuver is applied. The Origin field determines the origin of the maneuver coordinate system. The ability to define the coordinate system locally avoids having to create many coordinate systems, associated with specific spacecraft, in order to perform finite maneuvers for multiple spacecraft.	
	Default	Earth
	Limits	Any celestial body, libration point, or barycenter
	Units	N/A
Axes	The Axes field, together with the Origin field, describe the coordinate system in which a finite maneuver is applied. If VNB is chosen for Axes, a local coordinate system is created such that the x-axis points in the velocity direction of the spacecraft, with respect to the point defined by Origin, the y-axis points in the normal direction of the spacecraft with respect to Origin, and the z-axis completes the right-handed set.	
	Default	VNB
	Limits	VNB, MJ2000Eq
	Units	N/A
Thrusters	The Thrusters field allows the selection of which thrusters to use when applying a finite maneuver. The user can select more than one thruster, from the list of thrusters previously created, by including all thrusters in curly braces. An example is MyFiniteBurn.Thrusters = Thruster1,Thruster2,Thruster3.	
	Default	No Default
	Limits	Any thruster created by user
	Units	N/A
BurnScaleFactor	The BurnScaleFactor is used to scale the total acceleration before adding the acceleration due to a finite burn into the sum of the accelerations of a spacecraft. The scaling is performed by taking the sum of the accelerations applied by all thrusters specified under the Thrusters field, and multiplying the total thrust by BurnScaleFactor.	
	Default	1.0
	Limits	Real Number
	Units	N/A

Interactions

Spacecraft	A spacecraft must be created in order to apply any burn.
Thruster	Any thruster created in the resource tree can be incorporated into a finite burn to be used on the spacecraft.
Begin and End Finite Burn command	After a finite burn is created, to apply it to the mission sequence, a Begin and End Finite Burn command must be appended to the mission tree.

Examples

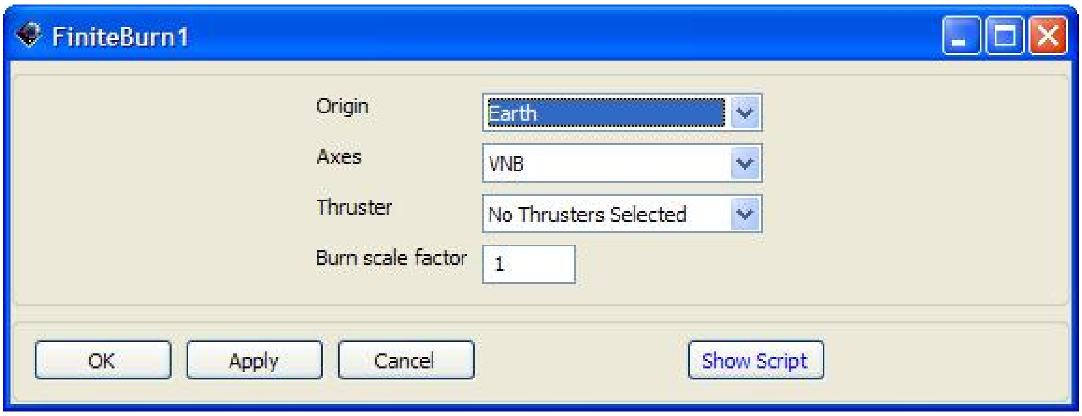


Figure: Default Name and Settings for the Finite Burn Object Dialog Box

```
Create FiniteBurn FiniteBurn1;
GMAT FiniteBurn1.Origin = Earth;
GMAT FiniteBurn1.Axes = VNB;
GMAT FiniteBurn1.BurnScaleFactor = 1;
```

Name

Formation — An ephemeris file.

Synopsis

```
Under Construction.
```

Description

Under Construction.

Fields

Field Name Description....

Type	Fill This In.
Default	Fill This In.
Limits	Fill This In.
Units	Fill This In.

Examples

Example 5. Example Script

```
%
```

Name

FuelTank — A fuel tank.

Synopsis

```
Create FuelTank name
name.field = value
```

Description

A FuelTank is a thermodynamc model of a tank and is required for finite burn modelling or for implusive burns that use mass depletion. The thermodynamic properties of the tank are modelled using the ideal gas law and assume that there is no energy transfer into or out of the tank as fuel is depleted. To use a FuelTank, you must first create the tank, and then attach it to the desired spacecraft and associate it with a thruster as shown in the examples below.

When working in the script, you must add tanks to spacecraft before the begin mission sequence command.

Fields

Pressure

The pressure in the tank.

Type	Real Number
Default	1500
Limits	Pressure > 0
Units	kPa.

Temperature

The temperature of the fuel and ullage in the tank. GMAT currently assumes ullage and fuel are always at the same temperature.

Type	Real Number
Default	20
Limits	Temperature > -273.15
Units	C.

FuelMass

The FuelMass field is the mass of fuel in the tank.

> **Caution**
>
> By default, GMAT will not allow the fuel mass to be negative. However, occasionally in iterative processes such as targeting, a solver will try values of a maneuver parameter that result in total fuel depletion. Using the default tank settings this will throw an excpeption stopping the run unless you set the AllowNegativeFuelMass flag to true.

Type	Real Number
Default	756
Limits	FuelMass > 0
Units	kg.

ReferenceTemperature

The temperature of the tank when fuel was loaded.

Type	Real Number
Default	20
Options	ReferenceTemperature >=; 0

	Units C.
Volume	The volume of the tank. GMAT checks to ensure that the volume of the tank is larger than the volume of fuel loaded in the tank and throws an exception in the case that the fuel volume is larger than the tank volume. Type Real Number Default 0.75 Options Real Number > 0 such that fuel volume is < tank volume. Units m^3.
FuelDensity	The density of the fuel. Type Real Numer Default 1260 Limits Real Number > 0 Units kg/m^3.
PressureModel	The pressure model describes how pressure in the tank changes as fuel is depleted. Type Enumeration Default PressureRegulated Limits PressureRegulated, BlowDown Units N/A
AllowNegativeFuelMass	This field allows the fuel tank to have negagive fuel mass which can be useful in optimization and targeting sequences before convergences has occurred. Default false Options true, false. Units N/A

Examples

Example 6. Creating a default FuelTank and attaching it to a Spacecraft

```
%  Create the Fuel Tank Object
Create FuelTank aTank;
aTank.AllowNegativeFuelMass = false;
aTank.FuelMass = 756;
aTank.Pressure = 1500;
aTank.Temperature = 20;
aTank.RefTemperature = 20;
aTank.Volume = 0.75;
aTank.FuelDensity = 1260;
aTank.PressureModel = PressureRegulated;

%  Create a Thruster and assign it a FuelTank
Create Thruster aThruster;
aThruster.Tank = {aTank};

%  Add the FuelTank and Thruster to a Spacecraft
Create Spacecraft aSpacecraft
aSpacecraft.Tanks = {aTank};
aSpacecraft.Thrusters = {aThruster};
```

Name

GroundStation — Under Construction.

Synopsis

```
Under Construction.
```

Description

Under Construction.

Fields

Field Name	Description....	
	Type	Fill This In.
	Default	Fill This In.
	Limits	Fill This In.
	Units	Fill This In.

Examples

Example 7. Example Script

```
%
```

Name

GroundTrack Plot — A GroundTrack Plot.

Synopsis

```
Create GroundTrackPlot name
name.field = value
```

Description

A GroundTrackPlot is a graphical display of the locus of subsatellite latitude and longitude points. The GroundTrackPlot in GMAT allows you to view a spacecraft's subsatellite point as illustrated by a spacecraft icon and the label for the spacecraft. Similarly, GroundStation locations are indicated with a ground station icon and label. The GroundTrackPlot object can display the ground track for multiple spacecraft simultaneously and can animate the ground track evolution after a GMAT run is complete. Like other graphical dispaly objects in GMAT, you can control how data is written to a ground track plot in iterative processes.

Fields

Fields Associated with Plot Options

DataCollectFrequency	The `DataCollectFrequency` field allows you to select a subset of the ephemeris data for drawing to a GroundTrackPlot. It is often inefficient to draw every ephemeris point associated with a trajectory and drawing a subset of the data provides a smooth groundtrack plot with faster execution times. The `DataCollectFrequency` is an integer that represents how many ephemeris points to skip between plotted data points in a GroundTrackPlot. If `DataCollectFrequency` is set to 10, then data is collected every 10 integration steps. Default 1 Limits Integer ≥ 1 Units Integration Steps
UpdatePlotFrequency	The UpdatePlotFrequency field allows you to specify the number of ephemiris data points to collect before updating a GroundTrackPlot with new latitude and longitude data. Data is collected every N propagation steps where N is defined by DataCollectFrequency. After M points are collected, where M is defined by UpdatePlotFrequency, the GroundTrackPlot is updated with new data. For example, if UpdatePlotFrequency is set to 10 and DataCollectFrequency is set to 2, then the plot is updated with new data every 20 (10*2) integration steps. Default 50 Limits Integer ≥ 1 Units Integration Steps
NumPointsToRedraw	When NumPointsToRedraw is set to zero, all collected ephemeris points are drawn. When NumPointsToRedraw is set to a positive integer, say 10 for example, only the last 10 collected data points are drawn. See DataCollectFrequency and UpdatePlotFrequency for an explanation of how data is collected for a GroundTrackPlotx.

	Default	0
	Limits	Integer ≥ 0
	Units	Integration Steps
ShowPlot	The ShowPlot field allows you to turn off the GroundTrackPlot display window without deleting the plot object or removing it from the script. If you select true, then the plot will be displayed. If you select false, then the plot will not be displayed.	
	Default	true
	Limits	true , false
	Units	N/A

Fields Associated with Drawing Options

Add	The Add field allows you to add Spacecraft and GroundStations to a GroundTrackPlot.	
	Default	DefaultSC, Earth
	Limits	SpacecraftName CelestialBodyName
	Units	N/A
Central Body	The CentralBody field allows you to specify the central body of a GroundTrackPlot. Currently, GMAT	
	Default	Earth
	Limits	CelestialBodyName
	Units	N/A

Fields Associated with Other Options

SolverIterations	The SolverIterations field determines if and how perturbed trajectories are drawn to a GroundTrackPlot during iterative a solver sequences. When SolverIterations is set to All, all solver iterations perturbations and iterations are shown on the plot. When SolverIterations is set to Current, only the current solver pass is shown on the plot and the iteration history is not retained. When SolverIterations is set to None, no perturbations or iterations are drawn and the GroundTrackPlot is not updated until the solver has converged.	
	Default	Current
	Limits	All, Current, None
	Units	N/A
TextureMap	The TextureMap field allows you to define a custom map file for use in a GroundTrackPlot.	
	Default	Current
	Limits	Texture map in jpg or bmp file.
	Units	N/A

Additional Information

When working in the GroundTrackPlot GUI, if you change the CentralBody field, the TextureMap field will automatically change to the default texture map for the new central body. If you have specified a custom texture map file and path, that information will be lost when you change the CentralBody field.

Interactions

Spacecraft	Any spacecraft in your mission is available to a GroundTrackPlot for display.

GroundStations	Any GroundSation in your mission is available to a GroundTrackPlot for display.
PenUp/PenDown Commands	You can use the PenUp and PenDown commands to control when data is written to a GroundTrackPlot
Toggle Command	You can use the Toggle command to control when data is written to a GroundTrackPlot

Examples

```
Create GroundTrackPlot GroundTrackPlot1;
GMAT GroundTrackPlot1.CentralBody = Earth;
GMAT GroundTrackPlot1.Add = {Sat, Earth};
GMAT GroundTrackPlot1.SolverIterations = Current;
GMAT GroundTrackPlot1.DataCollectFrequency = 1;
GMAT GroundTrackPlot1.UpdatePlotFrequency = 50;
GMAT GroundTrackPlot1.NumPointsToRedraw = 0;
GMAT GroundTrackPlot1.ShowPlot = true;
GMAT GroundTrackPlot1.TextureMap = '../MyMaps/MyTexture.jpg';
GMAT GroundTrackPlot1.UpperLeft = [ 0 0 ];
GMAT GroundTrackPlot1.Size = [ 0 0 ];
GMAT GroundTrackPlot1.RelativeZOrder = 0;
```

Name

Impulsive Burn — A impulsive burn.

Synopsis

```
Create ImpulsiveBurn name
name.field = value
```

Description

The impulsive burn object in GMAT allows the spacecraft to undergo an instantaneous ΔV in up to three dimensions as opposed to a finite burn which is not instantaneous. The user can configure the burn by defining its origin, type of axes, vector format, and magnitude of the vectors. Depending on the mission, it will be simpler to use one axes or vector format over the other.

Possible Coupling with Other Objects

Spacecraft	Must be created in order to apply any burn. The purpose of the impulsive burn is to instantaneously propel the spacecraft to either target or optimize a goal during its mission.
Maneuver command	Must be created to call the burn into the mission sequence because without a maneuver, the spacecraft simply propagates around a specified trajectory. If there are several burns that exist, in the Maneuver dialog box the user can choose which burn to utilize for that part of the mission sequence. In addition, a Propagate command must follow the maneuver to allow the trajectory to unfold after a burn has been applied.
Vary command	Required a burn to be specified in the Variable Setup group box. The purpose of the Vary command is to apply a burn in order to change a parameter of the spacecraft's trajectory.

Fields

Origin

Together the Origin and Axes fields describe the coordinate system in which a maneuver is applied. The Origin field determines the origin of the maneuver coordinatesystem. The ability to define the coordinate system locally avoids having to create many coordinate systems, associated with specific spacecraft, in order to perform finite maneuvers for multiple spacecraft.

Default	Earth
Limits	Any celestial body
Units	N/A

Axes

The Axes field, together with the Origin field, describethe coordinate system in which an impulsive maneuver is applied. If VNB is chosen for Axes, a local coordinate system is created such that the x-axis points in the velocity direction of thespacecraft, with respect to the point defined by Origin, the y-axis points in the normal direction of the spacecraft with respect to Origin, and the z-axis completes the right-handed set.

Default	VNB
Limits	VNB, MJ2000Eq
Units	N/A

VectorFormat	The VectorFormat field allows the user to define the format of the maneuver vector. Default Cartesian Limits Cartesian Units N/A
Element1	The Element1 field allows the user to define the first element of the impulsive maneuver vector. Element1 is X if VectorFormat is Cartesian. Default 0 Limits Real Number Units km/sec
Element2	The Element2 field allows the user to define the second element of the impulsive maneuver vector. Element2 is Y if VectorFormat is Cartesian. Default 0 Limits Real Number Units km/sec
Element3	The Element3 field allows the user to define the third element of the impulsive maneuver vector. Element3 is Z if VectorFormat is Cartesian. Default 0 Limits Real Number Units km/sec

Examples

```
Create ImpulsiveBurn ImpulsiveBurn1;
GMAT ImpulsiveBurn1.Origin = Earth;
GMAT ImpulsiveBurn1.Axes = VNB;
GMAT ImpulsiveBurn1.VectorFormat = Cartesian;
GMAT ImpulsiveBurn1.Element1 = 0;
GMAT ImpulsiveBurn1.Element2 = 0;
GMAT ImpulsiveBurn1.Element3 = 0;
```

Name

Libration Point — A libration point.

Synopsis

```
Create LibrationPoint name
name.field = value
```

Description

A Libration point, also called a Lagrange point, is a point of equilibrium in the restricted three-body problem.

Fields

Primary	The Primary field allows you to define the body treated as the primary in the calculation of the libration point location. (See Math. Spec for more details).	
	Default	Sun
	Limits	Sun, Mercury, Venus, Earth, Luna, Mars, Jupiter, Saturn, Uranus, Neptune, Pluto, or any Barycenter. (The Primary and Secondary bodies cannot be the same)
	Units	N/A
Secondary	The Secondary field allows you to define the body treated as the secondary in the calculation of the libration point location.	
	Default	Earth
	Limits	Sun, Mercury, Venus, Earth, Luna, Mars, Jupiter, Saturn, Uranus, Neptune, Pluto, or any Barycenter. (The Primary and Secondary bodies cannot be the same)
	Units	N/A
Point	The Point field specifies which libration point the object corresponds to.	
	Default	L1
	Limits	L1, L2, L3, L4, L5
	Units	N/A

Examples

Script Syntax

```
Create Libration Point;
.Primary = ;
.Secondary = ;
.Point = <'L1', 'L2', 'L3', 'L4', 'L5'>
```

Sample Script

```
Create LibrationPoint Libration1;
GMAT Libration1.Primary = Sun;
GMAT Libration1.Secondary = Earth;
GMAT Libration1.Point = 'L1';
```

Name

MATLABFunction — Under Construction.

Synopsis

```
Under Construction.
```

Description

Under Construction.

Fields

Field Name	Description....	
	Type	Fill This In.
	Default	Fill This In.
	Limits	Fill This In.
	Units	Fill This In.

Examples

Example 8. Example Script

```
%
```

Name

OpenGL Plot — A OpenGL Plot.

Synopsis

```
Create OpenGLPlot name
name.field = value
```

Description

Without OpenGL Plots, you would have no way of visualizing your spacecraft as it traveled along its trajectory. OpenGL Plots also have a multitude of options that allow you to customize your view of the spacecraft. This makes these types of plots very useful and in most cases necessary to using GMAT.

Fields

Fields Associated with Plot Options

DataCollectFrequency

The DataCollectFrequency field allows the user to define how data is collected for plotting. It is often inefficient to draw every ephemeris point associated with a trajectory. Often, drawing a smaller subset of the data still results in smooth trajectory plots, while executing more quickly. The DataCollectFrequency is an integer that represents how often to collect data and store for plotting. If DataCollectFrequency is set to 10, then Data is collected every 10 integration steps.

Default	1
Limits	Integer ≥ 1
Units	Integration Steps

UpdatePlotFrequency

The UpdatePlotFrequency field allows the user to specify how often to update an OpenGL plot is updated with new data collected during the process of propagating spacecraft and running a mission. Data is collected for a plot according the value defined by DataCollectFrequency. An OpenGL plot is updated with the new data, according to the value set in UpdatePlotFrequency. If UpdatePlotFrequency is set to 10 and DataCollectFrequency is set to 2, then the plot is updated with new data every 20 (10*2) integration steps.

Default	50
Limits	Integer ≥ 1
Units	Integration Steps

NumPointsToRedraw

When NumPointsToRedraw is set to zero, all ephemeris points are drawn. When NumPointsToRedraw is set to a positive integer, say 10 for example, only the last 10 collected data points are drawn. See DataCollectFrequency for explanation of how data is collected for an OpenGL plot.

Default	0
Limits	Integer ≥ 0
Units	Integration Steps

ShowPlot

The ShowPlot field allows the user to turn off a plot for a particular run, without deleting the plot object, or removing it from the script. If you select true, then the plot will be shown. If you select false, then the plot will not be shown.

Default	true
Limits	true , false
Units	N/A

Fields Associated with Viewed Objects

Add — The Add subfield adds a spacecraft,celestial body, libration point,or barycenter to a plot. When creating a plot the Earth is added as a default body and may be removed by using the Remove command. The user can add a spacecraft, celestial body, libration point, or barycenter to a plot by using the name used to create the object. The GUI's Selected field is the equivalent of the script's Add field. In the event of no Add command or no objects in the Selected field, GMAT should run without the OpenGL plot and a warning message displayed in the message window. The following warning message is sufficient: OpenGL plot will be turned off. No object has been selected for plotting.

Default	DefaultSC, Earth
Limits	SpacecraftName CelestialBodyName LibrationPointName BarycenterName
Units	N/A

Remove — The Remove subfield removes a spacecraft,celestial body, libration point, or barycenter from a plot. The user can remove any object that has been added to a plot by using the name used to add the object.

Default	No Default
Limits	Any object included in the Add list
Units	N/A

Fields Associated with Drawing Options

WireFrame — When the WireFrame field is set to On, celestial bodies are drawn using a wireframe model. When the WireFrame field is set to Off, then celestial bodies are drawn using a full map.

Default	Off
Limits	On , Off
Units	N/A

EclipticPlane — The EclipticPlane field allows the user to tell GMAT to draw a grid representing the ecliptic plane in an OpenGL plot. Note, the ecliptic plane can currently only be drawn for plots whose coordinate system uses the MJ2000Eq axis system.

Default	Off
Limits	On , Off Note: Only allowed for OpenGL plots with Coordinate Systems that use the MJ2000Eq axis system
Units	N/A

XYPlane — The XYPlane flag allows the user to tell GMAT to draw a grid representing the XY-plane of the coordinate system selected under the CoordinateSystem field of the OpenGL plot.

Default	On
Limits	On , Off
Units	N/A

Axes — The Axis flag allows the user to tell GMAT to draw the Cartesian axis system associated with the coordinate system selected under the CoordinateSystem field of an OpenGL plot.

	Default	On
	Limits	On , Off
	Units	N/A
Grid	The Grid flag allows the user to tell GMAT to draw a grid representing the longitude and latitude lines celestial bodies added to an OpenGL plot.	
	Default	On
	Limits	On , Off
	Units	N/A
EarthSunLines	The EarthSunLines allows the user to tell GMAT to draw a line that starts at the center of Earth and points towards the Sun.	
	Default	On
	Limits	On , Off
	Units	N/A
SolverIterations	The SolverIterations field determines whether or not perturbed trajectories are plotted during a solver (Targeter, Optimize) sequence. When SolverIterations is set to On, solver iterations are shown on the plot. When SolverIterations is Off, the solver iterations are not shown on the plot.	
	Default	Off
	Limits	On , Off
	Units	N/A

Fields Associated with View Definition

CoordinateSystem	The CoordinateSystem field on an OpenGL plot allows the user to select which coordinate system to use to draw the plot data. A coordinate system is defined as an origin and an axis system, and the CoordinateSystem field allows the user to determine the origin and axis system of an OpenGL plot. See the CoordinateSystem object fields for information of defining different types of coordinate systems.	
	Default	EarthMJ2000Eq
	Limits	Any default or user defined coordinate system
	Units	N/A
ViewPointReference	The ViewPointReference field is an optional field that allows the user to change the reference point from which ViewPointVector is measured. ViewPointReference} defaults to the origin of the coordinate system for the plot. A ViewPointReference can be any spacecraft, celestial body, libration point, or barycenter.	
	Default	Earth
	Limits	SpacecraftName, CelestialBodyName, LibrationPointName, BarycenterName, or a 3-vector of numerical values
	Units	N/A
ViewPointVector	The product of ViewScaleFactor and ViewPointVector field determines the view point location with respect to ViewPointReference. ViewPointVector can be a vector, or any of the following objects: spacecraft,celestial body, libration point,or barycenter. The location of the Viewpoint in three-space is defined as the vector addition of ViewPointReference, and the vector defined by product of ViewScaleFactor and ViewPointVector in the coordinate system chosen by the user.	

	Default	[0 0 30000]
	Limits	SpacecraftName, CelestialBodyName, LibrationPointName, BarycenterName, or a 3-vector of numerical values
	Units	km or N/A

ViewScaleFactor — The ViewScaleFactor field scales ViewPointVector before adding it to ViewPointReference. The ViewScaleFactor allows the user to back away from an object to fit in the field of view.

Default	1
Limits	Real Number ≥ 0
Units	N/A

ViewDirection — The ViewDirection field allows the user to select the direction of view in an OpenGL plot. The user can specify the view direction by choosing an object to point at such as a spacecraft,celestial body, libration point,or barycenter. Alternatively, the user can specify a vector of the form [x y z]. If the user specification of ViewDirection, ViewPointReference, and ViewPointVector, results in a zero vector, GMAT uses [0 0 10000] for ViewDirection.

Default	Earth
Limits	SpacecraftName, CelestialBodyName, LibrationPointName, BarycenterName, or a 3-vector of numerical values
Units	km or N/A

Fields Associated with View Options

UseInitialView — The UseInitialView field allows the user to control the view of an OpenGL plot between multiple runs of a mission sequence. The first time a specific OpenGL plot is created, GMAT will automatically use the view as defined by the fields associated with View Definition, View Up Direction, and Field of View. However, if the user changes the view using the mouse, GMAT will retain this view upon rerunning the mission if UseInitialView is set to false. If UseInitialView is set to true, the view for an OpenGL plot will be returned to the view defined by the initial settings.

Default	On
Limits	On , Off
Units	N/A

Fields Associated with View Up Definition

ViewUpCoordinate System — The ViewUpCoordinateSystem and ViewUpAxis fields are used to determine which direction appears as up in an OpenGL plot and together with the fields associated the the View Direction, uniquely define the view. The fields associated with the View Definition allow the user to define the point of view in 3-space, and the direction of the line of sight. However, this information alone is not enough to uniquely define the view. We also must provide how the view is oriented about the line of sight. This is accomplished by defining what direction should appear as the up direction in the plot and is configured using the ViewUpCoordinateSystem field and the ViewUpAx-

	is field. The ViewUpCoordinateSystem allows the user to select a coordinate system to define the up direction. Most of the time this system will be the same as the coordinate system chosen under the CoordinateSystem field. Default EarthMJ2000Eq Limits Any default or user defined coordinate system Units N/A	
ViewUpAxis	The ViewUpAxis allows the user to define which axis of the ViewUpCoordinateSystem that will appear as the up direction in an OpenGL plot. See the comments under ViewUpCoordinateSystem for more details of fields used to determine the up direction in an OpenGL plot. Default Z Limits X , -X , Y , -Y , Z , -Z Units N/A	

Interactions

Spacecraft	Any spacecraft in your mission is available to the OpenGL Plot for display
Solar System	The Sun and all of the Planets may be plotted or referenced in the OpenGL Plot If you add any Barrycenters or Libration Points, they will also be available for plotting and reference
Coordinate Systems	Both View Definition and View Up Definition may use the three default or user added coordinate systems

Examples

```
Create OpenGLPlot DefaultOpenGL;
GMAT DefaultOpenGL.SolverIterations = Current;
GMAT DefaultOpenGL.Add = {DefaultSC, Earth};
GMAT DefaultOpenGL.OrbitColor = [ 255 32768 ];
GMAT DefaultOpenGL.TargetColor = [ 8421440 0 ];
GMAT DefaultOpenGL.CoordinateSystem = EarthMJ2000Eq;
GMAT DefaultOpenGL.ViewPointReference = Earth;
GMAT DefaultOpenGL.ViewPointVector = [ 0 0 30000 ];
GMAT DefaultOpenGL.ViewDirection = Earth;
GMAT DefaultOpenGL.ViewScaleFactor = 1;
GMAT DefaultOpenGL.ViewUpCoordinateSystem = EarthMJ2000Eq;
GMAT DefaultOpenGL.ViewUpAxis = Z;
GMAT DefaultOpenGL.CelestialPlane = Off;
GMAT DefaultOpenGL.XYPlane = On;
GMAT DefaultOpenGL.WireFrame = Off;
GMAT DefaultOpenGL.Axes = On;
GMAT DefaultOpenGL.Grid = Off;
GMAT DefaultOpenGL.SunLine = Off;
GMAT DefaultOpenGL.UseInitialView = On;
GMAT DefaultOpenGL.DataCollectFrequency = 1;
GMAT DefaultOpenGL.UpdatePlotFrequency = 50;
GMAT DefaultOpenGL.NumPointsToRedraw = 0;
GMAT DefaultOpenGL.ShowPlot = true;
```

Name

Propagator — A propagator.

Synopsis

```
Create Propagator name
name.field = value
```

Description

In GMAT, a Propagator is a combination of an integrator and a force model. Hence, a Propagator contains a physical model of the space environment that is used to model the motion of a spacecraft as it moves forwards or backwards in time (VOP formulation is not currently supported). You configure a Propagator by selecting among different numerical integrators and environment models to create a Propagator appropriate to the flight regime of your spacecraft during its mission. GMAT supports numerous numerical integrators as well as Force Models like point mass and non-spherical gravity, atmospheric drag (Earth), and solar radiation pressure.

To propagate spacecraft in GMAT, you first create and configure a Propagator object in the script or in the Resource Tree. Then, in the mission sequence, you create a Propagate command, the topic of another section, and select among previously existing Propagators and Spacecraft. Hence, a Propagator is different from a Propagate command: A Propagator is a resource and is found in the GUI under the resource tree, and a Propagate Event is configured under the Mission Tree and is how you instruct GMAT to propagate spacecraft.

Interfaces

The Propagator dialog box is illustrated below and contains two group boxes: the Integrator group and the Force Model group. This section discusses the items in each group on the Propagate Panel. It will present how to configure a propagator and discuss all possible user settable fields in detail.

Integrator Group

The Integrator group allows you to select and configure a numerical integrator appropriate to your application. You select the type of numerical integrator in the -+Type+- pull-down menu. After selecting the integrator type, the fields below the -+Type+- pull-down menu dynamically configure to allow you to set relevant parameters for the selected integrator type. All integrators except for Adams-Bashforth-Moulton (ABM) are configured using the same fields. The ABM integrator has the following additional fields: -+MinIntegrationerror+- and -+NomIntegrationerror+-.

Force Model Group

The Force Model group allows you to configure a force model appropriate to the flight regime of your application. The central body of propagation and error control method are also defined here. On a Propagator, GMAT classifies all celestial bodies into two mutually exclusive categories: Primary Bodies, and Point Masses. Primary bodies can have a complex force model that includes non-spherical gravity, drag, and magnetic field. Point mass bodies only have a point-mass gravitational force.

You can add a Primary Body by clicking the Select button in the Primary Bodies group box. Once you have added a Primary Body (or mulitiple bodies in future versions), the pull down menu allows you to configure the force model for each Primary Body. The text box, next to the Select button contains a list of all Primary Bodies so you can see which bodies are being treated with complex force models. In future versions, GMAT will support multiple primary bodies on a propagator allowing you to use a non-spherical gravity model for the Earth and Moon simultaneously.

Configuring certain fields in the Force Model group affects the availability of other fields. For example, if you remove all bodies from the Primary Bodies list, the Gravity Field, Atmosphere Model, and Magnetic Field groups are disabled. Similarly, in the Gravity Field group, the search button and the Model File field are only active if "Other" is selected in the -+Type+- pull-down. In the Atmosphere Model group, the Setup button is only active when -+MSISE90+- or -+JacchiaRoberts+- are selected in the -+Type+- pull-down.

GMAT allows you to define Solar flux properties if you select either the -+MSISE90+- or -+JacchiaRoberts+- atmosphere models. By selecting one of these models in the -+Type+- pull-down menu in the Atmosphere Model group, the Setup button is enabled. Clicking on the Setup button brings up the panel illustrated below. Here you can input Solar flux values. GMAT does not currently support flux files though future versions will.

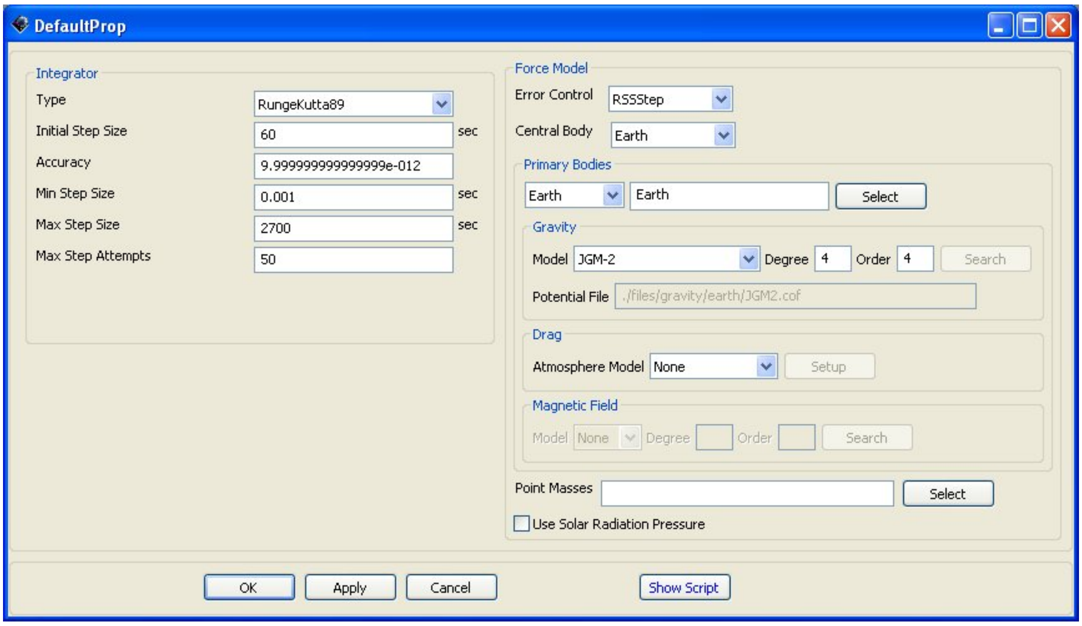


Figure: Default Name and Settings for the Propagator Object Dialog Box

Fields

Force Model Group Box Fields

ErrorControl	The ErrorControl field allows you to choose how a Propagator measures the error in an integration step. The algorithm selected in the ErrorControl field is used to determine the error in the current step, and this error is compared to the value set in the Accuracy field to determine if the step has an acceptable error or needs to be improved. All error measurements are relative error, however, the reference for the relative error changes depending upon the selection of ErrorControl. RSSStep is the Root Sum

Field	Description
	Square (RSS) relative error measured with respect to the current step. RSSState is the (RSS) relative error measured with respect to the current state. LargestStep is the state vector component with the largest relative error measured with respect to the current step. LargestState is the state vector component with the largest relative error measured with respect to the current state. For a more detailed discussion see the GMAT Mathematical Specification. Units: N/A. Default: RSSStep Limits: RSSStep, RSSState, LargestState, LargestStep
CentralBody	The CentralBody field allows the user to select the origin for the propagation. All propagation occurs in the FK5 axes system, about the CentralBody chosen by the user. The CentralBody must be a gravitational body and so cannot be a LibrationPoint or other special point. Default: Earth Limits: Sun, Mercury, Venus, Earth, Luna, Mars, Jupiter, Saturn, Uranus, Neptune, Pluto Units: N/A
PrimaryBodies	The PrimaryBodies field is a list of all celestial bodies that are to be modelled with a force model more complex than point mass gravity. Lists are surrounded by curly braces. For each PrimaryBody, the user can choose a drag, magnetic field, and aspherical gravity model. There is a coupling between the PrimaryBodies filed and the PointMasses field. A primary body can be any planet or moon not included in the PointMasses field. Default: Earth Limits: Sun, Mercury, Venus, Earth, Luna, Mars, Jupiter, Saturn, Uranus, Neptune, Pluto Units: N/A
Gravity.PrimaryBody.PotentialFile	This field allows the user to define the source for the non-spherical gravity coefficients for a primary body. If a gravity file is located in the Primary Body's potential path as defined in the startup file, you only need to specify the model name and not the entire path. For example, if the JGM2 coefficients file is contained in the directory defined in the startup file by the line EARTH_POT _PATH, then you only need to specify the model name JGM2. If the model is not contained in the body's potential path, you must supply the entire path as well as the file name. If GMAT does not successfully find the file requested, it uses the default gravity model as defined in the startup file. From the GUI, only models for Earth appear if Earth is the active primary body. This is to avoid allowing the user to select a lunar potential model for the Earth. If the Other option is selected the user has the ability of selecting a gravity model file on their local computer. Default: JGM2

	Limits	CentralBody-based models, Other. See Comments
	Units	N/A
Gravity.PrimaryBody.Degree	This field allows the user to select the the degree, or number of zonal terms, in the non-spherical gravity model. Ex. Gravity.Earth.Degree = 2 tells GMAT to use only the J2 zonal term for the Earth. The value for Degree must be less than the maximum degree specified by the Model.	
	Default	4
	Limits	Integer >= 0 and < the maximum specified by the model, Order <= Degree
	Units	N/A
Gravity.PrimaryBody.Order	This field allows the user to select the the order, or number of tesseral terms, in the non-spherical gravity model. Ex. Gravity.Earth.Order = 2 tells GMAT to use 2 tesseral terms. Note: Order must be greater than or equal to Degree.	
	Default	4
	Limits	Integer >= 0 and < the maximum specified by the model, Order <= Degree
	Units	N/A
Drag	The Drag field allows a user to specify a drag model. Currently, only one drag model can be chosen for a particular propagator and only Earth models are available.	
	Default	N/A
	Limits	None, JacchiaRoberts, MSISE90, Exponential
	Units	N/A. Note: This field will be deprecated in future versions of GMAT. Currently, the Drag field and the Drag.AtmosphereModel field must be set to the same value.
Drag.AtmosphereModel	The Drag.AtmosphereModel field allows a user to specify a drag model. Currently, only one drag model can be chosen for a particular propagator and only Earth models are available.	
	Default	None
	Limits	None, JacchiaRoberts, MSISE90, Exponential
	Units	N/A
Drag.F107	The Drag.F107 field allows you to set the $F_{10.7}$ solar flux value used in computing atmospheric density. $F_{10.7}$ is the solar radiation at a wavelength of 10.7 cm.	
	Default	150
	Limits	Real Number >= 0
	Units	$W/m^2/Hz$
Drag.F107A	The Drag.F107A field allows you to set the average $F_{10.7}$ value. $F_{10.7}$ is the average of $F_{10.7}$ over one month.	
	Default	150
	Limits	Real Number >= 0
	Units	$W/m^2/Hz$

Drag.MagneticIndex

The Drag.MagneticIndex index field allows you to set the k_p value for use in atmospheric density calculations. k_p is a planetary 3-hour-average, geomagnetic index that measures magnetic effects of solar radiation.

Default	3
Limits	0 <= Real Number <= 9
Units	N/A

PointMasses

A PointMass is a planet or moon that is modelled by a point source located at its center of gravity. A PointMass body can be any planet or moon not included in the PrimaryBodies field.

Default	None
Limits	Sun, Mercury, Venus, Earth, Luna, Mars, Jupiter, Saturn, Uranus, Neptune, Pluto
Units	N/A

SRP

The SRP field allows the user to include the force due to solar radiation pressure in the total sum of forces.

Default	Off
Limits	On, Off
Units	N/A

Integrator Group Box Fields

Type

The Type field is used to set the type of numerical integrator.

Default	RungeKutta89
Limits	RungeKutta89, RungeKutta68, RungeKutta56, PrinceDormand45, PrinceDormand78, BulirschStoer, AdamsBashforthMoulton

InitialStepSize

The InitialStepSize is the size of the first attempted step by the integrator. If the step defined by InitialStepSize does not satisfy Accuracy, the integrator adapts the step according an algorithm defined in the mathematical specifications document to find an acceptable first step that meets the user's requested.

Default	60 (sec)
Limits	Real Number
Units	seconds

Accuracy

The Accuracy field is used to set the desired accuracy for an integration step. When you set a value for Accuracy, GMAT uses the method selected in ErrorControl field on the Force Model, to determine a metric of the accuracy. For each step, the integrator ensures that the accuracy, as calculated using the method defined by ErrorControl, is less than the limit defined by Accuracy. If an integrator exceeds MaxStepAttempts trying to meet the requested accuracy, and error message is thrown and propagation stops.

Default	1e-11
Limits	Real Number >= 0
Units	N/A

MinStep

The MinStep field is used to set the minimum allowable step size.

Default	.001 (sec)
Limits	Real Number > 0, MinStep <= MaxStep
Units	seconds

MaxStep

The MaxStep field is used to set the maximum allowable step size.

Default	2700.0 (sec)

	Limits	Real Number > 0, MinStep <= MaxStep
	Units	seconds
MaxStepAttempts	The MaxStepAttempts field allows the user to set the number of attempts the integrator takes to meet the tolerance defined by Accuracy.	
	Default	50
	Limits	Integer > 0
	Units	None

Fields Associated Only with Adams-Bashforth-Moulton Integrator

MinIntegrationerror	The MinIntegrationerror field is used by the ABM integrator (and other predictor-corrector integrators when implemented) as the desired integration error to be obtained when the step size is changed. Predictor-Corrector integrators adapt step size when the obtained integration error falls outside of the range of acceptable steps, as determined by the bounds set by the MinIntegrationerror and Accuracy fields. The integrator then applies an internal calculation to recompute the step size, attempting to hit the NomIntegrationerror, and restarts the integrator.	
	Default	1.0e-13
	Limits	Real Number > 0, MinIntegrationerror < NomIntegrationerror < Accuracy
	Units	None
NomIntegrationerror	The NomIntegrationerror field is used by the ABM integrator (and other predictor-corrector integrators when implemented) as the desired integration error to be obtained when the step size is changed. Predictor-Corrector integrators adapt step size when the obtained integration error falls outside of the range of acceptable steps, as determined by the bounds set by the MinIntegrationerror and Accuracy fields. The integrator then applies an internal calculation to recompute the step size, attempting to hit the NomIntegrationerror, and restarts the integrator.	
	Default	1.0e-11
	Limits	Real Number > 0, MinIntegrationerror , NomIntegrationerror , Accuracy
	Units	None

Interactions

A Propagator Requires Other Objects/Commands of Type: Force Model (Script Only). (Note: There are slight differences in how you configure a Propagator in the script and GUI and we refer you to the script example shown in the script section for details. Effort has been made to reduce any difference between the script and GUI.)

Examples

```
Create ForceModel DefaultProp_ForceModel;
DefaultProp_ForceModel.CentralBody = Earth;
DefaultProp_ForceModel.PrimaryBodies = {Earth};
DefaultProp_ForceModel.Drag = None;
DefaultProp_ForceModel.SRP = Off;
DefaultProp_ForceModel.ErrorControl = RSSStep;
DefaultProp_ForceModel.GravityField.Earth.Degree = 4;
DefaultProp_ForceModel.GravityField.Earth.Order = 4;
```

```
DefaultProp_ForceModel.GravityField.Earth.PotentialFile = 'JGM2.cof';

Create Propagator DefaultProp;
DefaultProp.FM = DefaultProp_ForceModel;
DefaultProp.Type = RungeKutta89;
DefaultProp.InitialStepSize = 60;
DefaultProp.Accuracy = 9.999999999999999e-012;
DefaultProp.MinStep = 0.001;
DefaultProp.MaxStep = 2700;
DefaultProp.MaxStepAttempts = 50;
```

Name

ReportFile — A ReportFile.

Synopsis

```
Create ReportFile name
name.field = value
```

Description

The ReportFile is a file where values and qualities of objects can be stored so that they can be viewed at a later time.

Interfaces

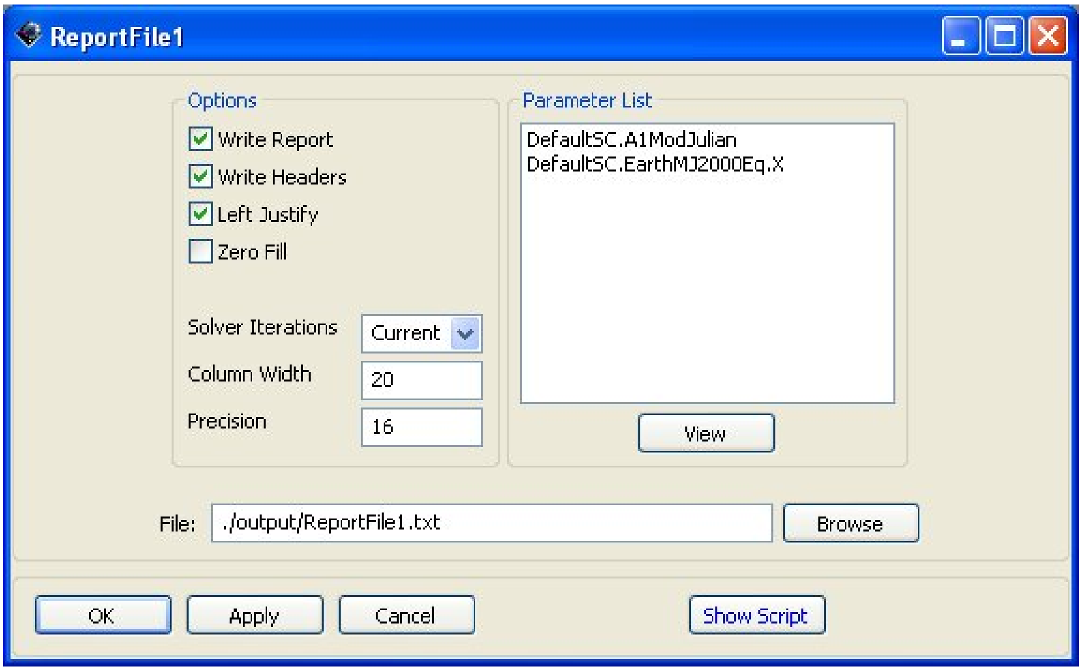


Figure: Default Name and Settings for the Report Object Dialog Box

Fields

FileName	The FileName field allows the user to define the file path and file name for a report.	
	Default	/RunReports/ReportFile1.txt
	Limits	Valid File Path and Name
	Units	None
Precision	The Precision field allows the user to set the precision of the variable written to a report.	
	Default	16
	Limits	Integer > 0
	Units	Same as variable being reported.
Add	The \st{Add} field allows a userto add user-defined variables to a report file. To add multipleuser-defined variables, enclose the variables with curly brackets.Ex. MyReportName.Add ={Sat.X, Sat.Y, Var1, Array(1,1)}; TheGUI's Selected field is the equivalent of the	

	script's Add field.In the event of no Add command or no objects in the Selected field, GMATshould run without the Report output and a warning message displayed in the message window. Thefollowing warning message is sufficient: Report plot will be turned off. No object has beenselected for reporting. Default None Limits Any user-defined parameter. Ex.Variables, Arrays, S/C parameters Units None	
WriteReport	The WriteReport field specifies whether to write data tothe report FileName. Default On Limits On,Off Units None	
WriteHeaders	The WriteHeaders field specifies whether to includeheaders that describe the variables in a report. Default On Limits On,Off Units None	
LeftJustify	When the LeftJustify field is set to On, then the data isleft justified and appears at the left most side of the column. Ifthe LeftJustify field is set to Off, then the data iscentered in the column. Default On Limits On, Off Units None	
ZeroFill	Default On Limits On, Off Units None	
ColumnWidth	The ColumnWidth field is used to define the width of the datacolumns in a report file. The value for ColumnWidth is appliedto all columns of data. For example, if ColumnWidth is set to20, then each data column will be 20 white-spaces wide. Default 20 Limits Integer > 0 Units Characters	
SolverIterations	The SolverIterations field determines whether or not data associated with perturbed trajectories during a solver (Targeter, Optimize) sequence is written to a report file. When SolverIterations is set to On, solver iterations are written to the report file. When SolverIterations is Off, the solver iterations are not written to the report file. Default Off Limits On, Off Units None	

Interactions

Report Command	Located in the mission tree and will retrieve values at that particular time and insert them at the bottom of the report file.

Examples

```
Create ReportFile ReportFile1;
ReportFile1.SolverIterations = Current;
```

```
ReportFile1.Filename = 'ReportFile1.txt';
ReportFile1.Precision = 16;
ReportFile1.Add = {DefaultSC.A1ModJulian, DefaultSC.EarthMJ2000Eq.X};
ReportFile1.WriteHeaders = On;
ReportFile1.LeftJustify = On;
ReportFile1.ZeroFill = Off;
ReportFile1.ColumnWidth = 20;
```

Name

Solar System — A solar system.

Synopsis

```
Create SolarSystem name
name.field = value
```

Description

This folder, found if the Solar System folder itself is double-clicked, enables the user to determine where he gets his data on the movements of planets, how often it updates, and how accurate the data is.

Fields

EphemerisSource	The EphemerisSourcefield allows the user to select the source used for planetaryephemerides. The source is used globally whenever planetaryephemeris information is required.	
	Default	DE405
	Limits	DE405,DE200, SLP, Analytic
	Units	None
Ephemeris UpdateInterval	The EphemerisUpdateInterval is used to set howoften planetary positions are updated when calculating accelerationsduring propagation. For low-Earth orbits,EphemerisUpdateInterval can be set to around 60 for fasternumerical integration with little effect on the accuracy of thepropagation. For deep space propagation,EphemerisUpdateInterval should be set to zero.	
	Default	0
	Limits	Real Number ≥ 0
	Units	sec
UseTTForEphemeris	GMAT uses time in the TDB system as thedefault time system in the JPL ephemeris files. However, often itis possible to use time in the TT time system, without significantdifference in propagation accuracy. (TT and TDB are within 1millisecond of each other). The advantage to using TT is that itavoids the transformation from TT to TDB and therefore orbitpropagation will execute faster. The UseTTForEphemeris fieldallows the user to choose between the default of TDB in theephemeris files (UseTTForEphemeris=false), or TT in theephemeris files (UseTTForEphemeris = true).	
	Default	false
	Limits	true, false
	Units	None
EphemerisFile	The EphemerisFile field allows the user to specify the locationand name of the file for each type of ephemeris GMAT supports. Forexample, if Ephemeris is set to DE405, you can set thepath for a	

	DE405 file using SolarSystem.EphemerisFile =c:/MyPath/MyDE405.file.	
	Default	Same as startup file.
	Limits	Filepath and file name consistent with operating system
	Units	None
AnalyticModel	Default	LowFidelity
	Limits	LowFidelity
	Units	None

Interactions

CelestialBodies, BaryCenter, and Libration Point	The position and data on all these depend on the source of the Solar System data and how often it is updated.
Propagator	How often the position of the planetary bodies are updated will have an impact on how a spacecraft will propagate.
Spacecraft	A number of parameters of a spacecraft are based off the position of the planets

Name

Spacecraft — A spacecraft

Synopsis

```
Create Spacecraft name
name.field = value
…
```

Description

A Spacecraft resource contains information about the spacecraft's orbit, its attitude, its physical parameters (such as mass and drag coefficient), and any attached hardware, including thrusters and fuel tanks. It also contains information about the visual model used to represent it in an OrbitView.

Fields

Epoch

`DateFormat` The entry format and time system of the Epoch field.

> **Caution**
>
> The definition of the modified Julian date is not the same as other software. Most software uses the Smithsonian Astrophysical Observatory definition of 1957, where JD is the full Julian date:
>
> MJD = JD - 2400000.5
>
> GMAT, however, uses the following definition:
>
> MJD = JD - 2430000.0

Value	Description
`A1Gregorian`	A.1 time scale, Gregorian format ("DD MMM YYYY hh:mm:ss.ddd")
`A1ModJulian`	A.1 time scale, modified Julian format
`TAIGregorian`	TAI time scale
`TAIModJulian`	*Default*
`TTGregorian`	TT time scale
`TTModJulian`	
`UTCGregorian`	UTC time scale
`UTCModJulian`	

`Epoch` The initial epoch of the spacecraft's state and properties.

Type	string
Default	`21545` (TAIModJulian format)
Limits	`04 Oct 1957 12:00:00.000` or later in all time systems (`6116.0` modified Julian)
Units	days (modified Julian format only)

Orbit

`CoordinateSystem` The Coordinate System field allows the user to choose which coordinate system with which to define the orbit state vector. The Coordinate System field has a dependency upon the State Type field. If the coordinate system chosen by the user does not have a gravitational body at the origin, then the state types Keplerian, ModifiedKeplerian, and Equinoctial are not permitted. This is because these state types require a μ value.

Type enumeration

Values
- `EarthMJ2000Eq` (default)
- `EarthMJ2000Ec`
- `EarthFixed`
- any user-defined coordinate system

`DisplayStateType` The State Type field allows the user to configure the type of state vector that they wish to use. The State Type field has a dependency upon the Coordinate System field. If the coordinate system chosen by the user does not have a gravitational body at the origin, then the state types Keplerian, ModifiedKeplerian, and Equinoctial are not permitted. This is because these state types require a μ value.

Type enumeration

Values
- `Cartesian` (default)
- `Keplerian`
- `ModifiedKeplerian`
- `SphericalAZFPA`
- `SphericalRADEC`
- `Equinoctial`

Cartesian State

`X` X is the x-component of the Spacecraft state in the coordinate system chosen in the Spacecraft Coordinate System field.

Type real number
Default 7100
Limits
Units km

`Y` Y is the y-component of the Spacecraft state in the coordinate system chosen in the Spacecraft Coordinate System field.

Type real number
Default 0
Limits
Units km

`Z` Z is the z-component of the Spacecraft state in the coordinate system chosen in the Spacecraft Coordinate System field.

Type real number
Default 1300
Limits
Units km

`VX` VX is the x-component of the Spacecraft velocity in the coordinate system chosen in the Spacecraft Coordinate System field.

Type real number
Default 0

Limits	
Units	km

`VY` VY is the y-component of the Spacecraft velocity in the coordinate system chosen in the Spacecraft Coordinate System field.

Type	real number
Default	7.35
Limits	
Units	km

`VZ` VZ is the z-component of the Spacecraft velocity in the coordinate system chosen in the Spacecraft Coordinate System field.

Type	real number
Default	1
Limits	
Units	km

Keplerian State

`SMA` The SMA field is the spacecraft orbit's osculating Keplerian semimajor axis in the coordinate system chosen in the Spacecraft Coordinate System field. SMA must be strictly greater than 1 m or less than -1 m to avoid numerical issues in the conversions to other state types. For circular and elliptical orbits (0 <= ECC < 0.9999999) SMA should only be greater than 1 m and for hyperbolic orbits (ECC > 1.0000001) SMA should be less than -1 m. GMAT does not support the creation of parabolic orbits.

Type	real number
Default	
Limits	|SMA| > 1e-3 km
Units	km

`ECC` The ECC field is the spacecraft orbit's osculating eccentricity. For circular or elliptic orbits, ECC must be greater than or equal to 0.0, and less than or equal to 0.9999999 to avoid numerical issues in the conversion to other state types as the Keplerian elements are undefined for parabolic orbits. For hyperbolic orbits ECC must be greater than or equal to 1.0000001. See also the SMA description.

Type	real number
Default	
Limits	0.0 <= ECC <= 0.9999999 or ECC >= 1.0000001
Units	none

`INC` The INC field is the spacecraft orbit's osculating inclination, in degrees, with respect to the selected coordinate system.

Type	real number
Default	
Limits	
Units	degrees

`RAAN` The RAAN field is the spacecraft orbit's osculating right ascension of the ascending node, in degrees, with respect to the selected coordinate system.

Type	real number
Default	
Limits	
Units	degrees

`AOP` The AOP field is the spacecraft orbit's osculating argument of periapsis, in degrees, with respect to the selected coordinate system.

Type	real number
Default	
Limits	

Units degrees

`TA` The TA field is the spacecraft orbit's osculating true anomaly.

Type real number
Default
Limits
Units degrees

ModifiedKeplerian State

`RadPer` The RadPer field is the spacecraft orbit's osculating radius of periapsis. RadPer must be greater than zero.

Type real number
Default
Limits RadPer > 0
Units km

`RadApo` The RadApo field is the spacecraft orbit's osculating radius of apoapsis. RadApo must be strictly greater than or less than zero. When RadApo is negative, the orbit is hyperbolic.

Type real number
Default
Limits RadApo ч 0
Units km

`INC` See the the section called “Keplerian State” section for a description of this field.

`RAAN` See the the section called “Keplerian State” section for a description of this field.

`AOP` See the the section called “Keplerian State” section for a description of this field.

`TA` See the the section called “Keplerian State” section for a description of this field.

SphericalAZFPA State

`RMAG` The RMAG field allows the user to set the magnitude of the spacecraft's position vector.

Type real number
Default
Limits RMAG > 0
Units km

`RA` The RA field allows the user to set the spacecraft's right ascension.

Type real number
Default
Limits
Units degrees

`DEC` The DEC field allows the user to set the spacecraft's declination.

Type real number
Default
Limits
Units degrees

`VMAG` The VMAG field allows the user to set the magnitude of the spacecraft's velocity.

Type real number
Default
Limits VMAG >= 0
Units km/s

`AZI` The AZI field allows the user to set the spacecraft's azimuth angle.

Type real number
Default

	Limits	
	Units	degrees
`FPA`	The FPA allows the user to set a spacecraft's flight path angle.	
	Type	real number
	Default	
	Limits	
	Units	degrees

Examples

Example 9. Creating a default Spacecraft

```
Create Spacecraft sc;
```

Name

SQP(fmincon) — A SQP(fmincon).

Synopsis

```
Create FminconOptimizer name
name.field = value
```

Description

fmincon is an Nonlinear Programming solver provided in MATLAB's Optimization Toolbox. fmincon performs nonlinear constrained optimization and supports linear and nonlinear constraints. This optimizer is only available to users who have both MATLAB and MATLAB's Optimization toolbox.

GMAT contains an interface to the fmincon optimizer and it appear as if fmincon is a built in optimizer in GMAT. Field names for this object have been copied from those used in MATLABS optimset function for consistency with MATLAB as opposed to other solvers in GMAT.

Interfaces

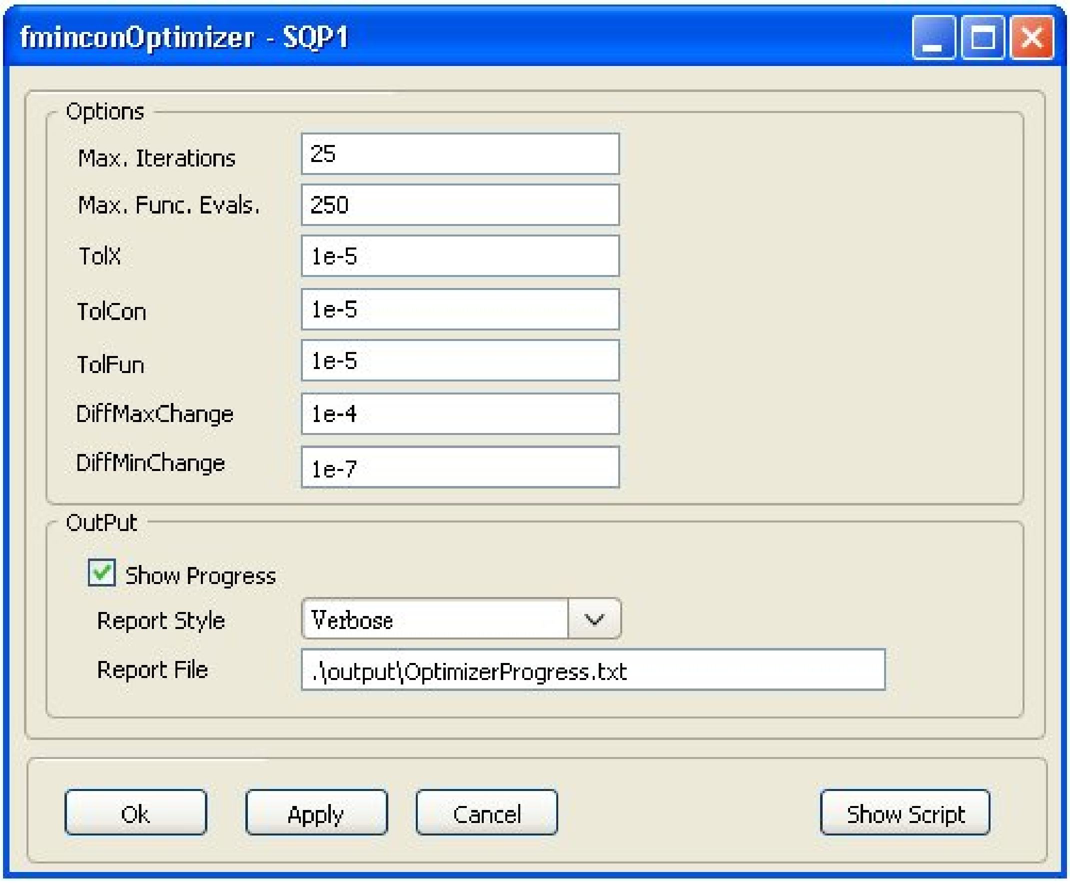


Figure: FminconOptimizer Dialog Box

Fields

DiffMax Change — The DiffMaxChange parameter is the upper limit on the perturbation used in MATLAB's finite differencing algorithm. For fmincon, you

	don't specify a single perturbation value, but rather give MATLAB a range, and it uses an adaptive algorithm that attempts to find the optimal perturbation. Default 0.1 Limits Real Number > 0 Units None
DiffMin Change	The DiffMinChange parameter is the lower limit on the perturbation used in MATLAB's finite differencing algorithm. For fmincon, you don't specify a single perturbation value, but rather give MATLAB a range, and it uses an adaptive algorithm that attempts to find the optimal perturbatin. Default 1e-8 Limits Real Number > 0 Units None
MaxFunEvals	The MaxFunEvals parameter allows the user to set the maximum number of cost function evaluations in an attempt to find an optimal solution. This is equivalent to setting the maximum number of passes through an optimization loop in a GMAT script. If a solution is not found before the maximum function evaluations, fmincon outputs an ExitFlag of zero, and GMAT continues. Default 1000 Limits Integer > 0 Units None
MaxIter	The MaxIter parameter allows the user to set the maximum allowable number of optimizer iterations. Depending upon the nature of the problem, and whether gradients are provided, it may take many function evaluations for each optimizer iteration. The MaxIter parameter allows the user to control the maximum function evaluations, and maximum iterations independently. Default 400 Limits Integer > 0 Units None
TolX	The TolX parameter is the termination toler-ance on the vector of independent variables, and is used only if the user sets a value. Default 1e-4 Limits Real Number > 0 Units None
TolFun	The TolFun parameter is the convergence tolerance on the cost function value. Default 1e-4 Limits Real Number > 0 Units None
TolCon	The TolCon parameter is the convergence tolerance on the constraint functions. Default 1e-4 Limits Real Number > 0 Units None
ShowProgress	The ShowProgress field determines whether data pertaining to iterations of the solver is displayed in the message window. When ShowProgress is true, the amount of information contained in the message window is controlled by the ReportSytle field. Default true

		Limits	true, false
		Units	None
ReportStyle	The ReportStyel field determines the amount and type of data written to the message window for each iteration of the solver (When ShowProgress is true). ADD DESCRIPTIONS OF CONCISE,VERBOSE, ADN NORMAL. I CAN'T RUN THE OPTIMIZER RIGHT NOW SO I CAN'T TELL WHAT EACH SETTING DOES.		
		Default	Normal
		Limits	Normal, Concise, Verbose, Debug
		Units	None
ReportFile	The ReportFile field contains the path and file name of the report file.		
		Default	Normal
		Limits	.\output\OptimizerData.txt
		Units	None

Using an fminconOptimizer

- Optimize Command
- Minimize
- Nonlinear Constraint

Examples

```
Create FminconOptimizer SQP1;
GMAT SQP1.MaxIter       = 25;
GMAT SQP1.MaxFunEvals   = 250;
GMAT SQP1.TolX          = 1e-5;
GMAT SQP1.TolFun        = 1e-5;
GMAT SQP1.TolCon        = 1e-5;
GMAT SQP1.DiffMaxChange = 1e-4;
GMAT SQP1.DiffMinChange = 1e-7;
GMAT SQP1.ShowProgress  = true;
GMAT SQP1.ReportStyle   = 'Verbose';
GMAT SQP1.ReportFile    = '\output\OptimizerProgress.txt';
```

Name

String — A string.

Synopsis

```
Create String name
name.field = value
```

Description

This page will show you how to create and use String objects. Stings are useful for storing characters as a set. One possible use of them is to report back a specific message at a set point in the mission sequence.

Interactions

ReportFile	You may add a string to the Parameter List in the Report Object dialog box via the ParameterSelectDialog box.
Report Command	Alternatively, you may add a string to the Parameter List in the Report Command dialog box

Using the Script

Script Syntax

```
GMAT String Name = String;
```

Script Examples

```
% The following is an example String declaration
GMAT Example = hello world;
```

Name

Thruster — A thruster.

Synopsis

```
Create Thruster name
name.field = value
```

Description

The Thruster uses the fuel tank and directs the thrust of the rocket engine while in space. It is used for finite burns.

Fields

CoordinateSystem	The CoordinateSystem field for a thrusterdetermines what coordinate system the orientation parametersX_Direction, Y_Direction, and Z_Direction arereferenced to. This is a temporary fix in GMAT. Eventually, theuser will specify the attitude of aspacecraft, and then X_Direction, Y_Direction, and Z_Direction will be referenced to the spacecraft body frame.	
	Default	EarthMJ2000Eq
	Limits	EarthMJ2000Eq, EarthMJ2000Ec, EarthMJ2000Eq, or anyuser defined system
Axis	The Axis field allows the user to define a localcoordinate system for a thruster. Note that there is a couplingbetween the Axis parameter and the CoordinateSystemparameter for a thruster. Only one of the two canbe specified.	
	Default	VNB
	Limits	InertialVNB
	Units	None
Origin	The Origin fieldallows the user to define a local origin for a thruster. Note thatthere is a coupling between the Origin parameter and theCoordinateSystem parameter for a thruster. Only one of the twocan be specified.	
	Default	Earth
	Limits	Sun, Mercury, Venus, Earth, Luna, Mars,Jupiter, Saturn, Uranus, Neptune, Pluto
	Units	None
X_Direction	X_Direction, divided by the RSS of the three directioncomponents, forms the x direction of the spacecraft thrust vector direction.	
	Default	1
	Limits	Real Number
Y_Direction	Y_Direction, divided by the RSS of the three directioncomponents, forms the y direction of the spacecraft thrust vector direction.	
	Default	0
	Limits	Real Number
Z_Direction	Z_Direction, divided by the RSS of the three directioncomponents, forms the z direction of the spacecraft thrust vector direction.	
	Default	0
	Limits	Real Number

ThrustScaleFactor	ThrustScaleFactor is a scale factor that is multiplied by the thrust vector for a given thruster, before the thrust vector is added into the total accleration.	
	Default	1
	Limits	Real Number > 0
	Units	None
Tank	The Tank field specifies which tank the thruster draws propellant from. The constants C_i below are used in the following equation to calculate thrust F_T as a function of pressure P and temperature T	
	Default	None
	Limits	Tank Name
	Units	$F_T(P,T) = \{C_1+C_2P+C_3P^2+C_4P^{C_5}+C_6P^{C_7}+C_8P^{C_9} +C_{10}C_{11}{}^{C_{12}P}\}(T/T_{ref})^{1+C_{13}+C_{14}P}$
C1	Thrust coefficient.	
	Default	500
	Limits	Real Number
	Units	N
C2	Thrust coefficient.	
	Default	0
	Limits	Real Number
	Units	N/kPa
C3	Thrust coefficient.	
	Default	0
	Limits	Real Number
	Units	N/kPa
C4	Thrust coefficient.	
	Default	0
	Limits	Real Number
	Units	N/kPa^{C5}
C5	Thrust coefficient.	
	Default	0
	Limits	Real Number
	Units	None
C6	Thrust coefficient.	
	Default	0
	Limits	Real Number
	Units	N/kPa^{C7}
C7	Thrust coefficient.	
	Default	0
	Limits	Real Number
	Units	None
C8	Thrust coefficient.	
	Default	0
	Limits	Real Number
	Units	N/kPa^{C9}
C9	Thrust coefficient.	
	Default	0
	Limits	Real Number
	Units	None
C10	Thrust coefficient.	
	Default	0

	Limits	Real Number
	Units	N
C11	Thrust coefficient.	
	Default	1
	Limits	Real Number
	Units	None
C12	Thrust coefficient.	
	Default	0
	Limits	Real Number
	Units	1/kPa
C13	Thrust coefficient.	
	Default	0
	Limits	Real Number
	Units	None
C14	Thrust coefficient.	
	Default	0
	Limits	Real Number
	Units	1/kPa
	The constants K_i below are used in the following equation tocalculate Isp as a function of pressure P and temperatureT	$I_{sp}(P,T)$ = $\{K_1+K_2P+K_3P^2+K_4P^{K_5}+K_6P^{K_7}+K_8P^{K_9}+K_{10}K_{11}^{K_{12}P}\}$ $(T/T_{ref})^{1+K_{13}+K_{14}P}$
K1	Isp coefficient.	
	Default	2150
	Limits	Real Number
	Units	m/sec
K2	Isp coefficient.	
	Default	0
	Limits	Real Number
	Units	m/(sec?kPa)
K3	Isp coefficient.	
	Default	0
	Limits	Real Number
	Units	m/(sec?kPa^2)
K4	Isp coefficient.	
	Default	0
	Limits	Real Number
	Units	m/(sec?kPa^{K5})
K5	Isp coefficient.	
	Default	0
	Limits	Real Number
	Units	None
K6	Isp coefficient.	
	Default	0
	Limits	Real Number
	Units	m/(sec?kPa^{K7})
K7	Isp coefficient.	
	Default	0
	Limits	Real Number
	Units	None
K8	Isp coefficient.	

	Default	0
	Limits	Real Number
	Units	m/(sec?kPa^{K9}
K9	Isp coefficient.	
	Default	0
	Limits	Real Number
	Units	None
K10	Isp coefficient.	
	Default	0
	Limits	Real Number
	Units	m/sec
K11	Isp coefficient.	
	Default	1
	Limits	Real Number
	Units	None
K12	Isp coefficient.	
	Default	0
	Limits	Real Number
	Units	1/kPa
K13	Isp coefficient.	
	Default	0
	Limits	Real Number
	Units	None
K14	Isp coefficient.	
	Default	0
	Limits	Real Number
	Units	1/kPa

Interactions

BeginFiniteBurn/EndFiniteBurn	These commands use the tank and the thruster to start a finite burn, where the delta V is not instantaneous.
Fuel Tank	This object contains the fuel used to power the thruster and subsequently the finite burn.
FiniteBurn	This takes the parameters of the tank and the thruster and apply it to a coordinate system, with a scaling method available if wanted.
Spacecraft	This is the object that the burn is applied to.

Script Examples

```
Create Thruster Thruster1;
GMAT Thruster1.Element1 = 1;
GMAT Thruster1.Element2 = 0;
GMAT Thruster1.Element3 = 0;
GMAT Thruster1.C1 = 500;
GMAT Thruster1.C2 = 0;
GMAT Thruster1.C3 = 0;
GMAT Thruster1.C4 = 0;
GMAT Thruster1.C5 = 0;
GMAT Thruster1.C6 = 0;
GMAT Thruster1.C7 = 0;
GMAT Thruster1.C8 = 0;
GMAT Thruster1.C9 = 0;
```

```
GMAT Thruster1.C10 = 0;
GMAT Thruster1.C11 = 1;
GMAT Thruster1.C12 = 0;
GMAT Thruster1.C13 = 0;
GMAT Thruster1.C14 = 0;
GMAT Thruster1.K1 = 2150;
GMAT Thruster1.K2 = 0;
GMAT Thruster1.K3 = 0;
GMAT Thruster1.K4 = 0;
GMAT Thruster1.K5 = 0;
GMAT Thruster1.K6 = 0;
GMAT Thruster1.K7 = 0;
GMAT Thruster1.K8 = 0;
GMAT Thruster1.K9 = 0;
GMAT Thruster1.K10 = 0;
GMAT Thruster1.K11 = 1;
GMAT Thruster1.K12 = 0;
GMAT Thruster1.K13 = 0;
GMAT Thruster1.K14 = 0;
GMAT Thruster1.CoordinateSystem = 'MJ2000EarthEquator';
GMAT Thruster1.ThrustScaleFactor = 1;
```

Name

Variable — A variable.

Synopsis

```
Create Variable name
name = value
```

Description

The Variable object allows you to create and name a variable and assign to it a real number value. A variable can be used in numerous commands which allows you to customize the mission sequence to your application. In the simplest case, a variable can be defined by a simple assignment to a numeric literal. In more complex cases, a variable can be defined using an assignment that contains a complicated mathematical expression.

Interfaces

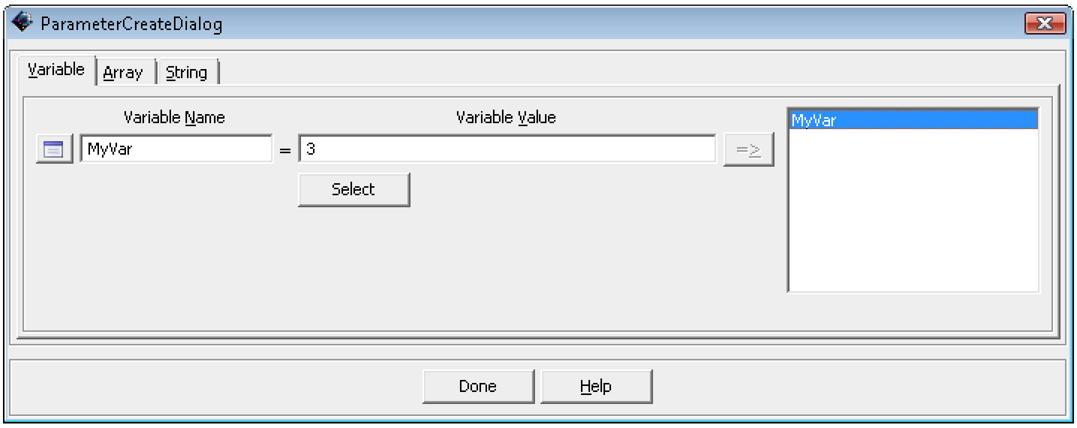


Figure: Parameter Create Dialog Box

Interactions

- Spacecraft Object
- Report Command
- Equation Command
- Vary Command
- Achieve Command
- Minimize Command

Examples

```
Create Variable pi Energy mu SMA
mu     = 398600.4415;
pi     = 3.14159265358979;
Energy = MySpacecraft.VMAG^2/2 - mu/r;
SMA    = -mu/2/Energy;
```

Name

VF13adOptimizer — Under Construction.

Synopsis

```
Under Construction.
```

Description

Under Construction.

Fields

Field Name	Description....	
	Type	Fill This In.
	Default	Fill This In.
	Limits	Fill This In.
	Units	Fill This In.

Examples

Example 10. Example Script

```
%
```

Name

XY Plot — A XY Plot.

Synopsis

```
Create XYPlot name
name.field = value
```

Description

The XY Plot is a graph under the Plots/Reports folder in the resource tree that plots data onto the X and Y axes of the graph. Any two parameters can be chosen to plot from the Parameter Select dialog box when the View radio button is clicked. The plot has the capability to be turned on and/or off throughout the mission if desirable.

Interactions

Spacecraft	Spacecraft interact with an XY Plot throughout the entire mission. The data retrieved from the spacecraft, as it carries out the command, is what gets plotted onto the graph.
Toggle Command	A Toggle can be inserted into the mission sequence to control when the XY Plot is to plot data by subscribing it to the Toggle list. If it is necessary to only plot data at a certain point during the mission, besides at the beginning or end points, then two Toggle commands can be added to switch it on and off.

Fields

IndVar

The IndVar field allows the user to define the independent variable for an xy-plot. Only one variable can be defined as an independent variable. For example, the line MyXYPlot.IndVar = DefaultSC.A1ModJulian sets the independent variable to be the epoch of DefaultSC in the A1 time system and modified Julian format.

Default	DefaultSC.A1ModJulian
Limits	Any user variable, array element, or spacecraft parameter
Units	None

Add

The Add field allows the user to add dependent variables to an xy-plot. All dependent variables are plotted on the y-axisvs the independent variable defined by IndVar. To define multiple dependent variables, they should be included in curly braces. For example, MyXYPlot.Add = DefaultSC.EarthMJ2000Eq.Y , DefaultSC.EarthMJ2000Eq.Z. The GUI's Selected field is the equivalent of the script's Add field. In the event of no Add command or no objects in the Selected field, GMAT should run without the XYPlot and a warning message displayed in the message window. The following warning message is sufficient: XYPlot will be turned off. No object has been selected for plotting.

Default	DefaultSC.EarthMJ2000Eq.X
Limits	Any user variable, array element, or spacecraft parameter

	Units	None
Grid	When the Grid field is set to On, then a grid is drawn on an xy-plot. When the Grid field is set to Off, then a grid is not drawn.	
	Default	On
	Limits	On, Off
	Units	None
SolverIterations	The SolverIterations field determines whether or not perturbed trajectories are plotted during a solver (Targeter, Optimize) sequence. When SolverIterations is set to On, solver iterations are shown on the plot. When SolverIterations is set to Off, solver iterations are not shown on the plot.	
	Default	Off
	Limits	On, Off
	Units	None
ShowPlot	The ShowPlot field allows the user to turn off a plot for a particular run, without deleting the plot object, or removing it from the script. If you select true, then the plot will be shown. If you select false, then the plot will not be shown.	
	Default	true
	Limits	true, false
	Units	None

Examples

```
Create XYPlot XYPlot1;
GMAT XYPlot1.SolverIterations = Current;
GMAT XYPlot1.IndVar = Sat.A1ModJulian;
GMAT XYPlot1.Add = {Sat.EarthMJ2000Eq.X};
GMAT XYPlot1.Grid = On;
GMAT XYPlot1.ShowPlot = true;
```

Commands

Name

Achieve — Perform an achieve command

Synopsis

Description

The purpose of the Achieve command is to define a goal for the spacecraft to reach at some point in its trajectory. The goal must have a corresponding value and tolerance so the differential corrector can solve for the best solution during the spacecraft's flight. To define a goal, a property must be chosen out of the Parameter Select dialog box along with the correct components in the other fields. The command can only be appended within a targeting sequence and must be accompanied and preceeded by a Vary, Maneuver, and Propagate command.

Options

Goal	The option allows the user to select any single element user defined parameter, except a number, to Achieve.	
	Default	DefaultSC.Earth.RMAG
	Limits	Spacecraft parameter, Array element, Variable, or any other single element user defined parameter, excluding numbers
Arg1	The Arg1 option is the desired value for after the solver has converged.	
	Default	42165
	Limits	Real Number, Array element, Variable, or any user defined parameter that obeys the conditions of Chapter~\\ref \{Ch:ObjectsNResources\} for the selected
	Units	None
Tolerance	The Tolerance option sets Arg2. Arg2 is the convergence tolerance for Arg1.	
	Default	0.1
	Limits	Real Number, Array element, Variable, or any user defined parameter > 0
	Units	None
SolverName	The SolverName option allows the user to choose which solver to assign to the Achieve command.	
	Default	DefaultDC
	Limits	Any user defined differential corrector
	Units	None

Examples

```
Achieve SolverName(Goal) = Arg1, Tolerance = Arg2
```

Name

BeginFiniteBurn — Perform a begin finite burn

Synopsis

Description

The Begin Finite Burn and End Finite Burn commands are very simple. When the Begin Finite Burn command is entered into the mission sequence it will initiate the thrusters of the spacecraft until the End Finite Burn command is reached. After the finite burn is turned off, the spacecraft's thrusters will shut down.

Options

ManeuverName	The ManeuverName option allows the user to choose between any previously created finite burn. As an example, to maneuver DefaultSC using DefaultFB, the script linewould appear as Manevuer DefaultFB(DefaultSC).	
	Default	DefaultFB
	Limits	Any finite burn existing in the resource tree or created in the script
	Units	None
SpacecraftName	The SpacecraftName option allows the user to select which spacecraft to maneuver using the maneuver selected in the ManeuverName option.	
	Default	DefaultSC
	Limits	Any spacecraft existing in the resource tree or created in the script
	Units	None

Examples

```
BeginFiniteBurn ManeuverName (SpacecraftName);
EndFiniteBurn DefaultFB(DefaultSC);
```

Name

BeginMissionSequence — Under construction.

Synopsis

```
Under construction.
```

Under construction.

Under construction.

Examples

Script Syntax

```
Under construction.
```

Script Examples

```
Under construction.
```

Name

Call Function — Perform a call function

Synopsis

Definition

GMAT functions are very useful and work nearly the same as they do in most programming languages. They may be invoked using the Call Function command covered here.

Options

OutputList	The OutputList option allows the user to set the output of Function to a user defined parameter.	
	Default	None
	Limits	Variables, Arrays, S/C, Paramters, any other user-defined parameters, or blank. Multiple outputs must be expressed in a comma delimited list format
	Units	None
InputList	The InputList option allows the user to set the input of Function to a user defined parameter.	
	Default	None
	Limits	Variables, Arrays, S/C, Paramters, any other user-defined parameters, or blank. Multiple inputs must be expressed in a comma delimited list format.
	Units	None
Function	The Function option allows the user to set the function that will be called in a specific location of the mission sequence. The function has to be defined before it can be used in the CallFunction Command.	
	Default	None
	Limits	GMAT of Matlab Function
	Units	None

Examples

Script Syntax

```
%Function call with Inputs and Outputs
GMAT [OutputList] = Function(InputList)

%Function call with Outputs only
GMAT [OutputList] = Function

%Function call with Inputs only
GMAT Function(InputList)

Function call with no Inputs or Outputs
GMAT Function
```

Script Examples

```
% Matlab function call without inputs or outputs
```

```
% Syntax 1
GMAT clearAll

% Syntax 2
GMAT [ ] = clearAll( )
```

Name

Else — Perform an else statement

Synopsis

Description

If-Else statements in GMAT work as they do in other programming languages, especially Matlab. The Else command adds another dimension to an If statement. You use an Else statement when you want something to happen when the conditions of an If statement are not met. For example, an If statement who's condition is "x < 5" will only execute the script within its scope when x is indeed less than 5. GMAT would otherwise pass over the If statement's associated script and continue. However, having an Else statement after the If will ensure that the lines of script within the scope of that Else are executed when x is equal to 5 or greater.

Examples

Script Syntax

```
If <logical expression>;
   <Statements>;
Else;
   <Statements>;
EndIf;
```

Script Examples

```
If DefaultSC.ElapsedDays < 1;
   Propagate DefaultProp( DefaultSC , { DefaultSC.ElapsedDays = 0.01 });
Else;
   Propagate DefaultProp( DefaultSC , { DefaultSC.ElapsedDays = 0.2 });
EndIf;
```

Name

EndFiniteBurn — Under construction.

Synopsis

```
Under construction.
```

Under construction.

Under construction.

Examples

Script Syntax

```
Under construction.
```

Script Examples

```
Under construction.
```

Name

Equation — Perform an equation command

Synopsis

Description

The Equation command uses the Equation to make one variable equal to some combination of previously defined variables and values. It is highly useful for storing values so that they aren't lost. Additionally, it is very useful for advanced commands.

Options

Arg1 The Arg1 option allows the user to set Arg1 to Arg2.

Default	None
Limits	Spacecraft Parameter, Array element, Variable, or any other single element user defined parameter
Units	None

Arg2 The Arg2 option allows the user to define Arg1.

Default	None
Limits	Spacecraft Parameter, Array element, Variable, any other single element user defined parameter, or a combination of the aforementioned parameters using math operators
Units	None

Examples

Script Syntax

```
GMAT Arg1 = Arg2;
```

Script Examples

```
% Setting a variable to a number
 GMAT testVar = 24;
 % Setting a variable to the value of a math statement
 GMAT testVar = (testVar2 + 50)/2;
```

Name

For — Perform a for loop

Synopsis

Description

The for loop is a control flow statement that allows portions of code to be executed iteratively using an explicit loop variable (Wikipedia). GMAT for loops are three-expression loops that allow the user to set the initial value of the loop variable, its increment, and the test to exit the loop. A parameter must be defined explicitly using a Create Variable statement or GUI equivalent before it can be used in a for loop command statement. The parameters used to define Start, Increment, and End can be any of the following GMAT parameters: numeric literal (real number), variable, array element, object property.

Interfaces

The GUI for the For Loop command is divided into four sections.

- The first section, the index, is where the counter variable name is displayed.
- The second section, the start, is the number with which the counter variable is first stored with.
- The third section, the increment, is the value that the counter variable will change by each time the program goes through the loop.
- The fourth section, the end, is the value of the counter variable when the loop is exited.

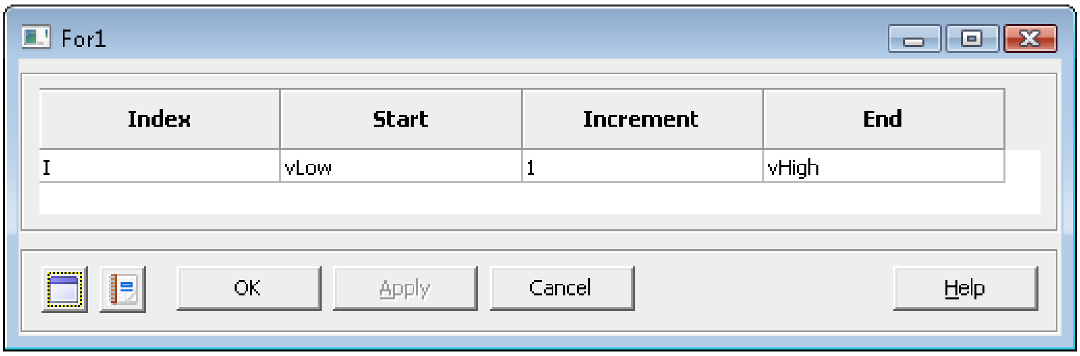


Figure: Default Name and Settings for the For Loop command Dialog Box

Script Syntax

(Simple For Loop)

```
For Variable = Start:End;

EndFor; }
```

(Expanded For Loop)

```
For Variable = Start:Increment:End;

EndFor;
```

Script Examples

```
% Output the value of the For loop Variable to a file
```

```
For I = 1:1:10;
GMAT testVar = I;
Report DefaultReportFile I;
EndFor;
```

Options

Variable	The Variable field allows the user to define the variable that to be incremented during the loop process.	
	Default	None
	Limits	Variable
	Units	None
Start	The Start option allows the user to set the starting value of the For Loop.	
	Default	1
	Limits	Real Number, Array element, Variable, or any user defined parameter
	Units	None
Increment	The Increment option allows the user to set the increment value of the For Loop. When the increment value is not included in a for loop statement, the default value is used.	
	Default	1
	Limits	Real Number, Array element, Variable, or any user defined parameter
	Units	None
End	The End option allows the user to set the ending value of the For Loop.	
	Default	10
	Limits	Real Number, Array, Variable, or any user defined parameter
	Units	None

For Loop Behavior

When the Increment option is left out of the script syntax the default value is used. If an Increment value of 0 is used, the For Loop should not execute but GMAT should continue to run. If End>Start and Increment<0, then the For Loop should not execute. If Start>End and Increment>0, then the For Loop should not execute. End can be equal to Start, but the For Loop will not execute.

Name

If — Perform an if command

Synopsis

Description

The If command gives you the ability to use a logical statement within GMAT. At some point during a mission sequence, when a particular command should only be executed when a certain condition is met, use of the If command is recommended. The If command also gives you the ability to make a command's execution reliant upon multiple conditions.

Options

<If Command>	Arg1 and Arg2 can be any of the following: Real Number, Array element, Variable, Spacecraft Parameter or any other user defined parameter.	
	Default	DefaultSC.ElapsedDays < 1.0
	Limits	Arg1 < Arg2 and < can be >, <, >=, <=, ==, ~126~=
	Units	None
<Statements>	Default	None
	Limits	Any script line that can be in the mission sequence
	Units	None
|	The | option allows the user to set an OR operator in between <logical expression>s.	
	Default	None
	Limits	None
	Units	None
&	The & option allows the user to set an AND operator in between <logical expression>s	
	Default	None
	Limits	None
	Units	None

Examples

Using the If command in the script is quite simple. If you have ever programmed before in the higher level languages such as C, Matlab, or Java, GMAT will be very familiar. The statement reads like you see it basically: If the given statement after the 'If' is true, then execute the statement(s) following until the 'EndIf' is reached.

Script Syntax

- Simple If Statement

```
If <logical expression>;
   <Statements>;
EndIf;
```

- Compound If statement

```
If <logical expression> | <logical expression> & <logical expression>;
   <Statements>;
EndIf;
```

Script Examples

```
If DefaultSC.ElapsedDays < 1;
   Propagate DefaultProp( DefaultSC , { DefaultSC.ElapsedDays = 0.01 });
EndIf;

If MyVariable < MyArray(1,1);
   MyArray(1,1) = 5;
EndIf;

If DefaultSC.Earth.TA < MyArray(1,2);
   Propagate DefaultProp( DefaultSC );
EndIf;
```

Name

Maneuver — Perform a maneuver command

Synopsis

Description

The Maneuver command is placed in the mission tree and applies a selected impulsive burn to a selected spacecraft. A finite burn requires something else to be applied.

Options

BurnName	The BurnName field allows the user to choose between any previously created impulsive burn. As an example, to maneuver DefaultSC using DefaultIB, the script line would appear as Manevuer DefaultIB(DefaultSC).	
	Default	DefaultIB
	Limits	Any impulsive burn existing in the resource tree or created in the script
	Units	None
SpacecraftName	The SpacecraftName field allows the user to select which spacecraft to maneuver using the maneuver selected in the BurnName field.	
	Default	DefaultSC
	Limits	Any spacecraft existing in the resource tree or created in the script
	Units	None

Examples

Script Syntax

```
Maneuver BurnName (SpacecraftName);
```

Script Examples

```
% Impulsive Burn
Maneuver DefaultIB(DefaultSC);
```

Name

Minimize — Perform a minimize command

Synopsis

Description

The minimize command in GMAT allows variables to minimize by using a defined optimizer in optimize sequence.

Possible Coupling with Other Objects

- fmincon Object
  - Must set in order to use Minimize Command.
- Optimize Command
  - Must be defined in order to use minimize command in optimize sequence.

Options

OptimizerName	The OptimizerName option allows the user to specify which solver to use to minimize the cost function.	
	Default	SQP1
	Limits	Any existing fmincon solver
	Units	None
Arg	The Arg field allows the user to specify the function to be minimized upon convergence of the solver given by OptimizerName. Arg can be any of the following: Variable, Array element, or Spacecraft Parameter or any other 1x1 numeric user defined parameter.	
	Default	DefaultSC.ECC
	Limits	Variable, Spacecraft parameter, or Array element
	Units	None

Examples

Script Syntax

```
Minimize OptimizerName (Arg)
```

Script Examples

```
% Minimize the eccentricity of Sat, using fminconSQP
Minimize fminconSQP(Sat.ECC);

% Minimize the Variable DeltaV, using fminconSQP
Minimize fminconSQP(DeltaV);

% Minimize the first component of MyArray, using fminconSQP
Minimize fminconSQP(MyArray(1,1));
```

Name

NonLinearConstraint — Apply nonlinear constraint

Synopsis

Description

The nonlinear constraints in GMAT allows spacecraft properties, variable or array to apply constraint values, and also NonLinearConstraint can be created in optimize sequences. By using the fmincon optimizer, users can give various nonlinear constraints.

Possible Coupling with Other Objects

- Optimize Command
  - NonLinearConstraints are used in Optimize Command.
- Optimizers (Solvers)
  - Must set optimizer in order to apply NonLinearConstraints.

Options

OptimizeName	The OptimizerName option allows the user to specify which solver to use in satisfying nonlinear constraints. Default: SQP1 Limits: Any existing fmincon solver Units: None
{logical expression}	The logical expression field allows the user to specify the constraint to be satisfied upon convergence of the solver given by OptimizerName. Arg1 and Arg2 can be any of the following: Real Number, a 1-D Array (column vector),Array element, Variable, Spacecraft Parameter or any other numeric user defined parameter. If Arg1 is a 1-D Array, then Arg2 must be a 1-D Array with the same dimensions and vice-versa. Default: DefaultSC.SMA = 7000 Limits: Arg1 <= Arg2 where <= can be >= ; <= ; = Units: None

Examples

Script Syntax

```
NonLinearConstraint OptimizerName ({logical expression})
```

Script Examples

```
% Constrain the SMA of Sat to be 7000 km, using fminconSQP
NonLinearConstraint fminconSQP( Sat.SMA = 7000 );

% Constrain the SMA of Sat to be less than or equal to 7000 km, using fminconSQP
NonLinearConstraint fminconSQP( Sat.SMA <= 7000 );

% Constrain the SMA of Sat to be greater than or equal to 7000 km, using fminconSQP
NonLinearConstraint fminconSQP( Sat.SMA >= 7000a );
```

Name

Optimize — Perform an optimize command

Synopsis

Description

The optimize command in GMAT allows variables to optimize by using a solver fmincon object.

Possible Coupling with Other Objects

- fmincon Object
  - Must set in order to use Optimize Command.
- Minimize
  - Must be defined if you would like to minimize variable in an optimize sequence.
- Vary
- NonLinearConstraint

Options

SolverName	The SolverName field allows the user to choose between any previously created optimizer for use in an optimization sequence. For example, to begin a optimization sequence using DefaultSQP, the script is Optimize DefaultSQP.	
	Default	DefaultSQP
	Limits	Any existing optimizer
	Units	None
<Statements>	Default	None
	Limits	Any non-targeter and non-optimizer command lines used in the mission sequence, as well as the optimizer dependent command lines Vary, NonLinearConstraint, and Minimize.
	Units	None

Examples

Script Syntax

```
Optimize SolverName;
   <Statements>;
EndOptimize;
```

Script Examples

```
% Beginning and ending syntax for the Optimize command
Optimize DefaultDC;

EndOptimize;
```

Name

PenUp — Under construction.

Synopsis

```
Under construction.
```

Under construction.

Under construction.

Examples

Script Syntax

```
Under construction.
```

Script Examples

```
Under construction.
```

Name

PenDown — Under construction.

Synopsis

```
Under construction.
```

Under construction.

Under construction.

Examples

Script Syntax

```
Under construction.
```

Script Examples

```
Under construction.
```

Name

Propagate — Perform a propagate command

Synopsis

Description

The Propagate Command is a very important one and will be covered in this section. Basically, Propagate will take the given spacecraft and, using the Propagat"or" specified as its guide, make it travel until the given condition is met whether it be a location or something else such as elapsed time.

Reference Table

BackProp	The BackProp option allows the user to set the flag to enable or disable backwards propagation for all spacecraft in the SatListN option. The Backward Propagation GUI check box field stores all the data in BackProp. A check indicates backward propagation is enabled and no check indicates forward propagation. In the script, BackProp can be the word Backwards for backward propagation or blank for forward propagation.	
	Default	None
	Limits	Backwards or None
	Units	None
Mode	The Mode option allows the user to set the propagation mode for the propagator that will affect all of the spacecraft added to the SatListN option. For example, if synchronized is selected, all spacecraft are propagated at the same step size. The Propagate Mode GUI field stores all the data in Mode. In the script, Mode is left blank for the None option and the text of the other options available is used for their respective modes.	
	Default	None
	Limits	Synchronized or None
	Units	None
PropagatorName	The PropagatorName option allows the user to select a user defined propagator to use in spacecraft and/or formation propagation. The Propagator GUI field stores all the data in PropagatorName.	
	Default	DefaultProp
	Limits	Default propagator or any user-defined propagator
	Units	None
SatListN	The SatListN option allows the user to enter all the satellites and/or formations they want to propagate using the PropagatorName propagator settings. The Spacecraft List GUI field stores all the data in SatListN.	
	Default	DefaultSC
	Limits	Any existing spacecraft or formations, not being propagated by another propagator

		in the same Propagate event. Multiple spacecraft must be expressed in a comma delimited list format.
	Units	None
StopCondListN / Parameter	The StopCondListN option allows the user to enter all the parameters used for the propagator stopping condition. See the StopCondListN/Condition Option/Field for additional details to the StopCondListN option.	
	Default	DefaultSC.ElapsedSecs
	Limits	Any single element user accessible spacecraft parameter followed by an equal sign
	Units	None
StopCondListN / Condition	The StopCondListN option allows the user to enter the propagator stopping condition's value for the StopCondListN Parameter field.	
	Default	8640.0
	Limits	Real Number, Array element, Variable, spacecraft parameter, or any user defined parameter
	Units	Dependant on the condition selected.

Examples

Script Syntax

```
Propagate Mode BackProp PropagatorName(SatList1,fStopCondList1g) ...
BackPropPropagatorName (SatListN, {StopCondList})
```

Script Examples

```
% Single spacecraft propagation with one stopping condition
% Syntax #1
Propagate DefaultProp(DefaultSC, {DefaultSC.ElapsedSecs = 8640.0});

% Single spacecraft propagation with one stopping condition
% Syntax #2
Propagate DefaultProp(DefaultSC) {DefaultSC.ElapsedSecs = 8640.0};

% Single spacecraft propagation by one integration step
Propagate DefaultProp(DefaultSC);

% Multiple spacecraft propagation by one integration step
Propagate DefaultProp(Sat1, Sat2, Sat3);

% Single formation propagation by one integration step
Propagate DefaultProp(DefaultFormation);

% Single spacecraft backwards propagation by one integration step
Propagate Backwards DefaultProp(DefaultSC);

% Two spacecraft synchronized propagation with one stopping condition
Propagate Synchronized DefaultProp(Sat1, Sat2, {DefaultSC.ElapsedSecs = 8640.0});

% Multiple spacecraft propagation with multiple stopping conditions and propagation settin
% Syntax #1
Propagate Prop1(Sat1,Sat2, {Sat1.ElapsedSecs = 8640.0, Sat2.MA = 90}) ...
```

```
Prop2(Sat3, {Sat3.TA = 0.0});

% Multiple spacecraft propagation with multiple stopping conditions and propagation
% Syntax #2
Propagate Prop1(Sat1,Sat2) {Sat1.ElapsedSecs = 8640.0, Sat2.MA = 90} ...
Prop2(Sat3), {Sat3.TA = 0.0};
```

Name

Report — Output a report

Synopsis

Description

The report command allows the user find parameters of the orbit and the spacecraft at particular moments in time. This command is inserted into the mission tree at various locations in the mission tree. The parameters found by this command are placed into a report file that can be accessed at a later time.

Options

ReportName — The ReportName option allows the user to specift the ReportFile for data output.

Default	None
Limits	Any ReportFile created
Units	None

DataList — The DataList option allows the user to output data to the Filename specified by the ReportName. Multiple objects can be in the DataList when they are separated by spaces.

Default	None
Limits	Spacecraft parameter, Array, Variable, String, or any other single user defined parameter
Units	None

Examples

Script Syntax

```
Report  ReportName  DataList
```

Script Examples

```
%Report the time and position of DefaultSC

Report DefaultReport DefaultSC.A1ModJulian DefaultSC.X DefaultSC.Y DefaultSC.Z;
```

Name

Save — Under construction.

Synopsis

```
Under construction.
```

Under construction.

Under construction.

Examples

Script Syntax

```
Under construction.
```

Script Examples

```
Under construction.
```

Name

ScriptEvent — Perform a ScriptEvent command

Synopsis

Overview

The ScriptEvent command allows a user to enter in script within the GUI in the mission sequence. This can be useful in a number of ways if its easier to enter the script than to use the GUI interface. Also, if multiple things needed to be changed rapidly, this could be useful. Additionally, the ScriptEvent allows a user to have more freedom if the GUI is problematic and won't allow the user to perform the desired operation.

Options

<Statements>	Default	None
	Limits	Any valid line of GMAT script
	Units	None

Examples

Script Syntax

```
BeginScript
     <Statements>
EndScript;
```

Script Examples

```
% Assignment Command inside Script Event

BeginScript
     GMAT testVar = 24;
EndScript;
```

Name

Stop — Perform a stop command

Synopsis

Description

The Stop command simply ends a mission in GMAT. Whether placed in the script or GUI, Stop will halt the execution of the mission and have GMAT report back that the Command Sequence was intentionally interrupted.

Examples

Script Syntax

```
Stop
```

Script Examples

```
% Stop between propagation sequences
Propagate DefaultProp(DefaultSC) DefaultSC.ElapsedSecs = 8640.0;
Stop;
Propagate DefaultProp(DefaultSC) DefaultSC.ElapsedDays = 10.0;
```

Name

Target — Perform a targeting sequence

Synopsis

```
Target SolverName(options)
    Vary SolverName(Vary arguments);
    Achieve SolverName(Achieve arguments);
    [Other Commands;]
    …
EndTarget
```

Description

When building a mission, the targeting sequence is one of the most useful tools you have at your disposal. It allows you to use the power of a differential corrector to find solutions to complex two and three-body problems with a limited number of known values and several variables. Target is the base command that is appended with more commands such as Vary, Maneuver, Propagate, and Achieve in order to meet your goals. This makes it the first part of the sequence to be put into the Mission tree.

Arguments

SolverName — The SolverName option allows the user to choose between any previously created differential correctors for use in a targeting sequence. For example, to begin a targeting sequence using DefaultDC, the script is Target DefaultDC.

Type: DifferentialCorrector

Options

SolveMode — Tells the targeter state machine how to manage the targeter loop. When running in the RunInitialGuess mode, the target loop is executed one time using the initial values of the targeter variables. No targeting is performed. When running in Solve mode, the targeter searches for a solution satisfying the targeter goals.

Type: Enumeration

Values:
- Solve (default)
- RunInitialGuess

ExitMode — Tells the Mission Control Sequence how to proceed after the targeter loop has finished running. When running in DiscardAndContinue mode, subsequent calls to the same targeter loop will reset the variables to their initial values. If running in SaveAndContinue mode, the targeter variables used on a subsequent run start with the values obtained the last time the targeter loop was run. In Stop mode, the mission run halts when the targeter loop completes its work.

Type: Enumeration

Values:
- DiscardAndContinue (default)
- SaveAndContinue
- Stop

Examples

Example 11. Targeting geosynchronous orbit using an impulsive burn

```
Target DefaultDC
    Vary DefaultDC(DefaultIB.Element1 = 0.5);
    Maneuver DefaultIB(DefaultSC);
    Propagate DefaultProp(DefaultSC) {DefaultSC.Earth.Apoapsis};
    Achieve DefaultDC(DefaultSC.Earth.RMAG = 42165.0);
EndTarget
```

Name

Toggle — Perform a toggle command

Synopsis

Description

The Toggle command is useful in turning on/off certain plots so that they do not become unintelligible. They are particularly useful if a mission requires multiple sequences to fine-tune the answer. If this is the case, it might be better to turn off a plot until the final or second-to-last sequence so that the rougher sequences don't obstruct the view of what is going on in the finesse sequences.

Options

OutputNames	The Toggle option allows the user to assign the Plot/Report(s) to be toggled. When more than one Plot/Report is being toggled they need to be separated by a space.	
	Default	DefaultOpenGL
	Limits	Any OpenGL, Report, XYplot, or any other Plot/Report type
	Units	None
Arg	The Arg option allows the user to turn off or on the data output to a Plot/Report.	
	Default	On
	Limits	On or Off
	Units	None

Exmaples

Script Syntax

```
Toggle OutputNames Arg
```

Script Examples

```
% Turn off Report file for the first day of propagation
Toggle ReportFile1 Off
Propagate DefaultProp(DefaultSC, {DefaultSC.ElapsedDays = 1});
Toggle ReportFile1 On
Propagate DefaultProp(DefaultSC, {DefaultSC.ElapsedDays = 1});

% Turn off XYPlot and Report file for the first day of propagation
Toggle XYPlot1 ReportFile1 Off
Propagate DefaultProp(DefaultSC, {DefaultSC.ElapsedDays = 1});

Toggle XYPlot1 ReportFile1 On
Propagate DefaultProp(DefaultSC, {DefaultSC.ElapsedDays = 1})
```

Name

Vary — Perform a vary command

Synopsis

Description

The Vary command is used in conjunction with the Target and Optimize Commands. The Vary command varies a particular parameter within the Target and Optimize Commands until certain conditions are met. It is a highly useful command for creating missions. Multiple Vary commands can be used within the same targeting or optimizing sequence.

Options

Parameters Associated with All Solvers

SolverName — The SolverName option allows the user to choose which solver to assign to the vary command.

Default	DefaultDC
Limits	Any user defined solver
Units	None

Variable — The Variable option allows the user to select any single element user defined parameter, except a number, to vary. For example, DefaultIB.V, DefaultIB.N, DefaultIB.Element1, DefaultSC.TA, Array(1,1), and Variable are all valid values. The three element burn vector or multidimensional Arrays are not valid values.

Default	DefaultIB.V
Limits	Spacecrafy paremeter, Array element, Variable, or any other single element user defined parameter, excluding numbers
Units	None

InitialGuess — The InitialGuess option allows the user to set the initial guess for the selected Variable.

Default	0.5
Limits	Real Number, Array element, Variable, or any user defined parameter that obeys the conditions for the selected Variable object
Units	km/s

Lower — The Lower option allows the user to set Arg3 to the lower bound of the quantity being varied.

Default	0.0
Limits	Real Number, Array element, Variable, or any user defined parameter (Upper > Lower)
Units	None

Upper — The Upper option allows the user to set Arg4 to the upper bound of the quantity being varied.

Default	3.14159
Limits	Real Number, Array element, Variable, or any user defined parameter (Upper > Lower)
Units	None

Parameters Associated with Differential Corrector

Perturbation	The Perturbation option is set by specifying a value for Arg1. The value of Arg1 is the perturbation size in calculating the finite difference derivative. Default 1e-4 Limits Real Number, Array element, Variable, or any user defined parameter > 0 Units None	
MaxStep	The MaxStep option is set by specifying a value for Arg2. The value of Arg2 limits the size of the step taken during an interaction of the differential corrector. Default 0.2 Limits Real Number, Array element, Variable, or any user defined parameter > 0 Units None	

Parameters Associated with fmincon Optimizer

Additive Scale Factor	The AdditiveScaleFactor Field is used to nondimensionalize the independent variable. fmincon sees only the nondimensional form of the variable. The nondimensionalization is performed using the following equation: $x_n = (x_d - a)/m$. (x_n is the non-dimensional parameter. x_d is the dimensional parameter. a= additive scale factor. m= multiplicative scale factor.) Default 0 Limits Real Number, Array element, Variable, or any user defined parameter Units None
Multiplicative Scale Factor	The MultiplicativeScaleFactor Field is used to nondimensionalize the independent variable. fmincon sees only the nondimensional form of the variable. The nondimensionalization is performed using the following equation: $x_n = (x_d - a)/m$. (x_n is the non-dimensional parameter. x_d is the dimensional parameter. a= additive scale factor. m= multiplicative scale factor.) Default 1.0 Limits Real Number, Array element, Variable, or any user defined parameter Units None

Examples

Script Syntax

```
Vary SolverName(Variable=InitialGuess,{Perturbation=Arg1, MaxStep=Arg2,
  Lower=Arg3,...Upper=Arg4, AdditiveScalefactor=Arg6,MultiplicativeScalefactor=Arg6})
```

Script Examples

```
% Impulsive Burn Vary Command

Vary DefaultDC(DefaultIB.V = 0.5, {Perturbation = 0.0001, MaxStep = 0.2,
Lower = 0, Upper = 3.14159});
```

Name

While — Run a while loop

Synopsis

Description

The while loop is a control logic function that allows GMAT to check the spacecraft's status on a given parameter while performing a command or another control logic function within the mission sequence. When a spacecraft has reached the given property, the while loop will check its condition and react according to the equation defined in the loop's dialog box.

Options

<logical expression>

Arg1 and Arg2 can be any of the following: Real Number, Array, Variable, Spacecraft Parameter or any other user defined parameter.

- Default: DefaultSC.ElapsedDays < 1.0
- Limits: Arg1 < Arg2 and < can be > , < , >= , <= , == , ~=
- Units: None

<Statements>

- Default: None
- Limits: Any script line that can be in the mission sequence
- Units: None

|

The | option allows the user to set an OR operator in between s.

- Default: None
- Limits: None
- Units: None

&

The & option allows the user to set an AND operator in between <logical expression>s.

- Default: None
- Limits: None
- Units: None

Examples

Script Syntax

- Simple While Loop

```
While <logical expression>;
    <Statements>;
EndWhile;
```

- Compound While Loop

```
While <logical expression> | <logical expression> & <logical expression>;
    <Statements>
EndWhile;
```

Script Examples

```
While DefaultSC.ElapsedDays < 1;
```

```
Propagate DefaultProp (DefaultSC , DefaultSC.Elapsed Days = 0.01);
EndWhile;

While MyVariable < MyArray(1,1);
MyArray(1,1) = 5;
EndWhile;
```

Appendix A. NASA Open Source Agreement v1.3

```
NASA OPEN SOURCE SOFTWARE AGREEMENT
NASA OPEN SOURCE AGREEMENT VERSION 1.3

THIS OPEN SOURCE AGREEMENT ("AGREEMENT") DEFINES THE RIGHTS OF
 USE,
REPRODUCTION, DISTRIBUTION, MODIFICATION AND REDISTRIBUTION
 OF CERTAIN COMPUTER SOFTWARE ORIGINALLY RELEASED BY THE
 UNITED STATES GOVERNMENT AS REPRESENTED BY THE GOVERNMENT
 AGENCY LISTED BELOW ("GOVERNMENT AGENCY"). THE UNITED
 STATES GOVERNMENT, AS REPRESENTED BY GOVERNMENT AGENCY,
 IS AN INTENDED THIRD-PARTY BENEFICIARY OF ALL SUBSEQUENT
 DISTRIBUTIONS OR REDISTRIBUTIONS OF THE SUBJECT SOFTWARE.
 ANYONE WHO USES, REPRODUCES, DISTRIBUTES, MODIFIES OR
 REDISTRIBUTES THE SUBJECT SOFTWARE, AS DEFINED HEREIN, OR
 ANY PART THEREOF, IS, BY THAT ACTION, ACCEPTING IN FULL THE
 RESPONSIBILITIES AND OBLIGATIONS CONTAINED IN THIS AGREEMENT.
Government Agency: National Aeronautics and Space Administration
Government Agency Original Software Designation: GSC-16228-1
Government Agency Original Software Title: General Mission Analysis Tool (GMAT), version
 2011A
User Registration Requested. Please visit: http://opensource.gsfc.nasa.gov
Government Agency Point of Contact for Original Software: Dale Hithon, SRA Assistant,
 (301) 286-2691 NOSA Version 1.3 Page 2 of 6
1. DEFINITIONS
A. "Contributor" means Government Agency, as the developer of the Original Software, and
 any entity that makes a Modification.
B. "Covered Patents" mean patent claims licensable by a Contributor that are necessarily
 infringed by the use or sale of its Modification alone or when combined with the Subject
 Software.
C. "Display" means the showing of a copy of the Subject Software, either directly or by means
 of an image, or any other device.
D. "Distribution" means conveyance or transfer of the Subject Software, regardless of means,
 to another.
E. "Larger Work" means computer software that combines Subject Software, or portions
 thereof, with software separate from the Subject Software that is not governed by the terms
 of this Agreement.
F. "Modification" means any alteration of, including addition to or deletion from, the
 substance or structure of either the Original Software or Subject Software, and includes
 derivative works, as that term is defined in the Copyright Statute, 17 USC 101. However, the
 act of including Subject Software as part of a Larger Work does not in and of itself constitute
 a Modification.
G. "Original Software" means the computer software first released under this Agreement
 by Government Agency with Government Agency designation GSC-16228-1 and
 entitled General Mission Analysis Tool (GMAT), including source code, object code and
 accompanying documentation, if any.
H. "Recipient" means anyone who acquires the Subject Software under this Agreement,
 including all Contributors.
```

I. "Redistribution" means Distribution of the Subject Software after a Modification has been made.
J. "Reproduction" means the making of a counterpart, image or copy of the Subject Software.
K. "Sale" means the exchange of the Subject Software for money or equivalent value.
L. "Subject Software" means the Original Software, Modifications, or any respective parts thereof.
M. "Use" means the application or employment of the Subject Software for any purpose.
NOSA Version 1.3 Page 3 of 6
2. GRANT OF RIGHTS
A. Under Non-Patent Rights: Subject to the terms and conditions of this Agreement, each Contributor, with respect to its own contribution to the Subject Software, hereby grants to each Recipient a non-exclusive, world-wide, royalty-free license to engage in the following activities pertaining to the Subject Software:
1. Use
2. Distribution
3. Reproduction
4. Modification
5. Redistribution
6. Display
B. Under Patent Rights: Subject to the terms and conditions of this Agreement, each Contributor, with respect to its own contribution to the Subject Software, hereby grants to each Recipient under Covered Patents a non-exclusive, world-wide, royalty-free license to engage in the following activities pertaining to the Subject Software:
1. Use
2. Distribution
3. Reproduction
4. Sale
5. Offer for Sale
C. The rights granted under Paragraph B. also apply to the combination of a Contributor's Modification and the Subject Software if, at the time the Modification is added by the Contributor, the addition of such Modification causes the combination to be covered by the Covered Patents. It does not apply to any other combinations that include a Modification.
D. The rights granted in Paragraphs A. and B. allow the Recipient to sublicense those same rights. Such sublicense must be under the same terms and conditions of this Agreement.
NOSA Version 1.3 Page 4 of 6
3. OBLIGATIONS OF RECIPIENT
A. Distribution or Redistribution of the Subject Software must be made under this Agreement except for additions covered under paragraph 3H.
1. Whenever a Recipient distributes or redistributes the Subject Software, a copy of this Agreement must be included with each copy of the Subject Software; and
2. If Recipient distributes or redistributes the Subject Software in any form other than source code, Recipient must also make the source code freely available, and must provide with each copy of the Subject Software information on how to obtain the source code in a reasonable manner on or through a medium customarily used for software exchange.
B. Each Recipient must ensure that the following copyright notice appears prominently in the Subject Software:
Copyright © 2002 - 2011 United States Government as represented by the Administrator of the National Aeronautics and Space Administration. All Other Rights Reserved.
C. Each Contributor must characterize its alteration of the Subject Software as a Modification and must identify itself as the originator of its Modification in a manner that reasonably allows subsequent Recipients to identify the originator of the Modification. In fulfillment of these requirements, Contributor must include a file (e.g., a change log file) that describes the

alterations made and the date of the alterations, identifies Contributor as originator of the alterations, and consents to characterization of the alterations as a Modification, for example, by including a statement that the Modification is derived, directly or indirectly, from Original Software provided by Government Agency. Once consent is granted, it may not thereafter be revoked.

D. A Contributor may add its own copyright notice to the Subject Software. Once a copyright notice has been added to the Subject Software, a Recipient may not remove it without the express permission of the Contributor who added the notice.

E. A Recipient may not make any representation in the Subject Software or in any promotional, advertising or other material that may be construed as an endorsement by Government Agency or by any prior Recipient of any product or service provided by Recipient, or that may seek to obtain commercial advantage by the fact of Government Agency's
or a prior Recipient's participation in this Agreement.

F. In an effort to track usage and maintain accurate records of the Subject Software, each Recipient, upon receipt of the Subject Software, is requested to register with Government Agency by visiting the following website: http://opensource.gsfc.nasa.gov. Recipient's name and personal information shall be used for statistical purposes only. Once a Recipient makes a Modification available, it is requested that the Recipient inform Government Agency at the web site provided above how to access the Modification. NOSA Version 1.3 Page 5 of 6

G. Each Contributor represents that its Modification is believed to be Contributor's original creation and does not violate any existing agreements, regulations, statutes or rules, and further that Contributor has sufficient rights to grant the rights conveyed by this Agreement.

H. A Recipient may choose to offer, and to charge a fee for, warranty, support, indemnity and/or liability obligations to one or more other Recipients of the Subject Software. A Recipient may do so, however, only on its own behalf and not on behalf of Government Agency or any other Recipient. Such a Recipient must make it absolutely clear that any such warranty, support, indemnity and/or liability obligation is offered by that Recipient alone. Further, such Recipient agrees to indemnify Government Agency and every other Recipient for any liability incurred by them as a result of warranty, support, indemnity and/or liability offered by such Recipient.

I. A Recipient may create a Larger Work by combining Subject Software with separate software not governed by the terms of this agreement and distribute the Larger Work as a single product. In such case, the Recipient must make sure Subject Software, or portions thereof, included in the Larger Work is subject to this Agreement.

J. Notwithstanding any provisions contained herein, Recipient is hereby put on notice that export of any goods or technical data from the United States may require some form of export license from the U.S. Government. Failure to obtain necessary export licenses may result in criminal liability under U.S. laws. Government Agency neither represents that a license shall not be required nor that, if required, it shall be issued. Nothing granted herein provides any such export license.

4. DISCLAIMER OF WARRANTIES AND LIABILITIES; WAIVER AND INDEMNIFICATION

A. No Warranty: THE SUBJECT SOFTWARE IS PROVIDED "AS IS" WITHOUT ANY WARRANTY OF ANY KIND, EITHER EXPRESSED, IMPLIED, OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, ANY WARRANTY THAT THE SUBJECT SOFTWARE WILL CONFORM TO SPECIFICATIONS, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR FREEDOM FROM INFRINGEMENT, ANY WARRANTY THAT THE SUBJECT SOFTWARE WILL BE ERROR FREE, OR ANY WARRANTY THAT DOCUMENTATION, IF PROVIDED, WILL CONFORM TO THE SUBJECT SOFTWARE. THIS AGREEMENT DOES NOT, IN ANY MANNER, CONSTITUTE AN ENDORSEMENT BY

GOVERNMENT AGENCY OR ANY PRIOR RECIPIENT OF ANY RESULTS, RESULTING DESIGNS, HARDWARE, SOFTWARE PRODUCTS OR ANY OTHER APPLICATIONS RESULTING FROM USE OF THE SUBJECT SOFTWARE. FURTHER, GOVERNMENT AGENCY DISCLAIMS ALL WARRANTIES AND LIABILITIES REGARDING THIRD-PARTY SOFTWARE, IF PRESENT IN THE ORIGINAL SOFTWARE, AND DISTRIBUTES IT "AS IS."

B. Waiver and Indemnity: RECIPIENT AGREES TO WAIVE ANY AND ALL CLAIMS AGAINST THE UNITED STATES GOVERNMENT, ITS CONTRACTORS AND SUBCONTRACTORS, AS WELL AS ANY PRIOR RECIPIENT. IF RECIPIENT'S USE OF NOSA Version 1.3 Page 6 of 6

THE SUBJECT SOFTWARE RESULTS IN ANY LIABILITIES, DEMANDS, DAMAGES, EXPENSES OR LOSSES ARISING FROM SUCH USE, INCLUDING ANY DAMAGES FROM PRODUCTS BASED ON, OR RESULTING FROM, RECIPIENT'S USE OF THE SUBJECT SOFTWARE, RECIPIENT SHALL INDEMNIFY AND HOLD HARMLESS THE UNITED STATES GOVERNMENT, ITS CONTRACTORS AND SUBCONTRACTORS, AS WELL AS ANY PRIOR RECIPIENT, TO THE EXTENT PERMITTED BY LAW. RECIPIENT'S SOLE REMEDY FOR ANY SUCH MATTER SHALL BE THE IMMEDIATE, UNILATERAL TERMINATION OF THIS AGREEMENT.

5. GENERAL TERMS

A. Termination: This Agreement and the rights granted hereunder will terminate automatically if a Recipient fails to comply with these terms and conditions, and fails to cure such noncompliance within thirty (30) days of becoming aware of such noncompliance. Upon termination, a Recipient agrees to immediately cease use and distribution of the Subject Software. All sublicenses to the Subject Software properly granted by the breaching Recipient shall survive any such termination of this Agreement.

B. Severability: If any provision of this Agreement is invalid or unenforceable under applicable law, it shall not affect the validity or enforceability of the remainder of the terms of this Agreement.

C. Applicable Law: This Agreement shall be subject to United States federal law only for all purposes, including, but not limited to, determining the validity of this Agreement, the meaning of its provisions and the rights, obligations and remedies of the parties.

D. Entire Understanding: This Agreement constitutes the entire understanding and agreement of the parties relating to release of the Subject Software and may not be superseded, modified or amended except by further written agreement duly executed by the parties.

E. Binding Authority: By accepting and using the Subject Software under this Agreement, a Recipient affirms its authority to bind the Recipient to all terms and conditions of this Agreement and that that Recipient hereby agrees to all terms and conditions herein.

F. Point of Contact: Any Recipient contact with Government Agency is to be directed to the designated representative as follows: Dale Hithon, SRA Assistant, (301) 286-2691

Index