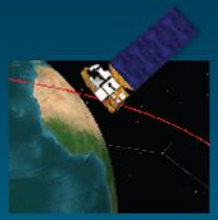
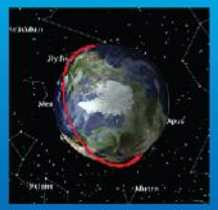
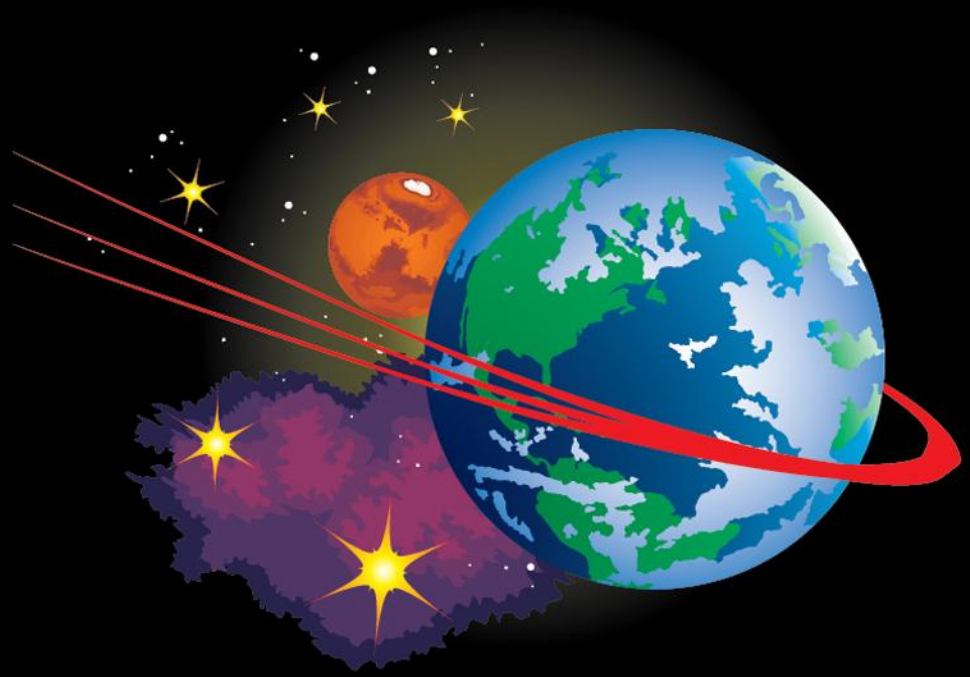
National Aeronautics and Space Administration





www.nasa.gov



Introduction to the GMAT Software

GMAT Fundamentals

Jason Laing and Mojtaba Abedin

Oct 29, 2014

NASA Goddard Space Flight Center

general mission analysis tool

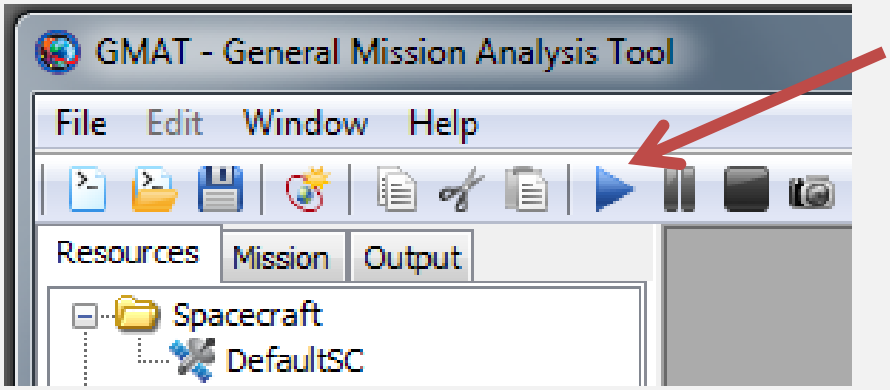
Outline

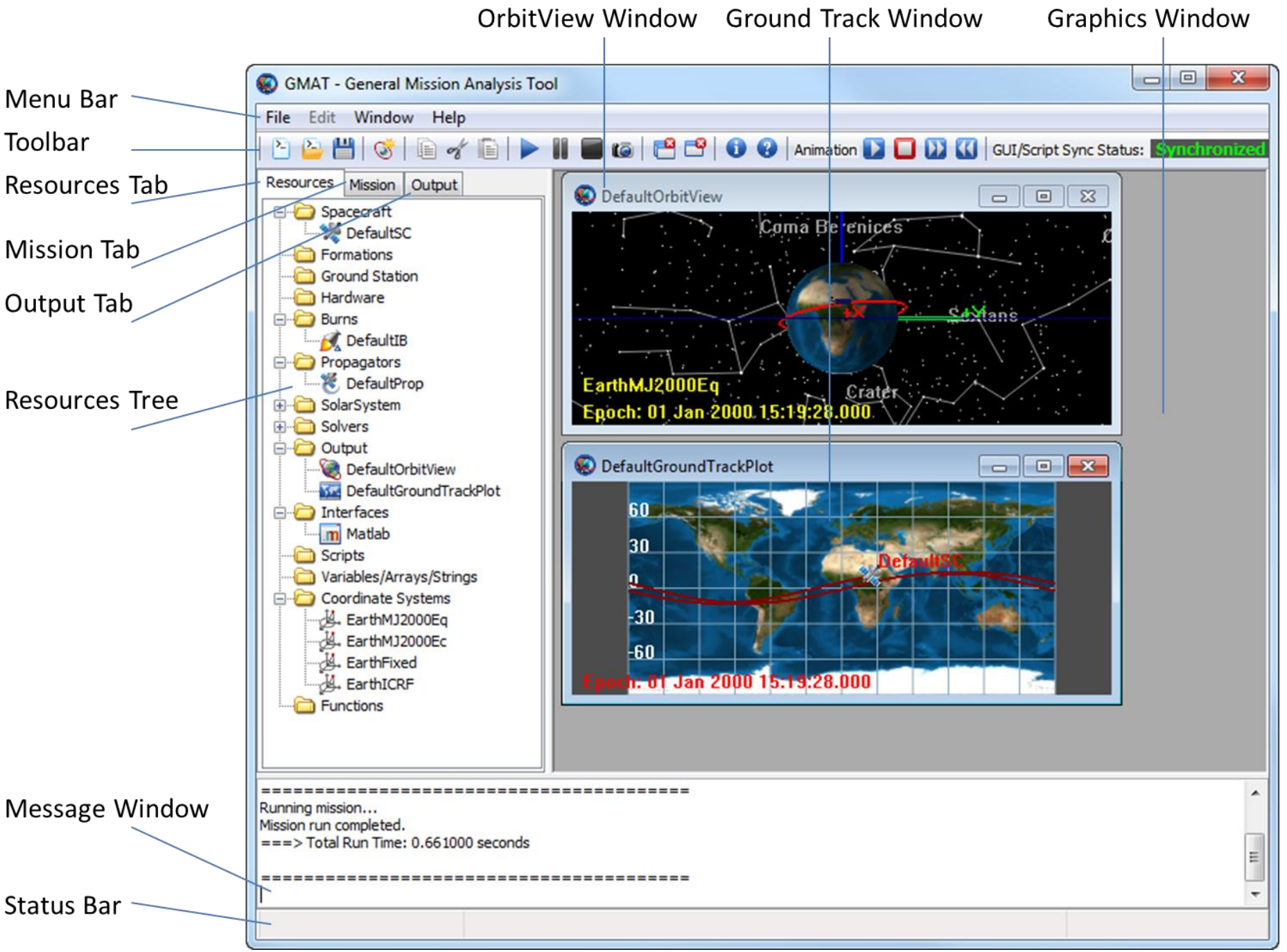
I. **Starting GMAT**
II. **Key Concepts**
   a. Two Parallel Interfaces
   b. Resources and Commands
   c. Fields and Parameters
   d. Execution Model
III. **Tour of the Graphical User Interface**
   a. GUI Controls
   b. Resources Tree
   c. Mission Tree
   d. Output Tree
   e. OrbitView
IV. **Tour of the Script Language**
   a. Basic Syntax
   b. Control Structures
   c. Using Math
   d. Using Parameters
   e. Solvers
   f. Script Editor
   g. Best Practices
V. **Data Files and Configuration**
VI. **Plugins**
VII. **Getting Help**

Starting GMAT

1. Double-click the icon:

2. Click **Run** to run the default mission:




OrbitView Window
Ground Track Window
Graphics Window
Menu Bar
Toolbar
Resources Tab
Mission Tab
Output Tab
Resources Tree
Message Window
Status Bar
GMAT - General Mission Analysis Tool
File Edit Window Help
Animation
GUI/Script Sync Status: Synchronized
Resources
Mission
Output
Spacecraft
DefaultSC
Formations
Ground Station
Hardware
Burns
DefaultIB
Propagators
DefaultProp
SolarSystem
Solvers
Output
DefaultOrbitView
DefaultGroundTrackPlot
Interfaces
Matlab
Scripts
Variables/Arrays/Strings
Coordinate Systems
EarthMJ2000Eq
EarthMJ2000Ec
EarthFixed
EarthICRF
Functions
DefaultOrbitView
Coma Berenices
EarthMJ2000Eq
Epoch: 01 Jan 2000 15:19:28.000
Crater
DefaultGroundTrackPlot
60
30
0
-30
-60
DefaultSC
Epoch: 01 Jan 2000 15:19:28.000
Running mission...
Mission run completed.
===> Total Run Time: 0.661000 seconds

KEY CONCEPTS

GMAT

Key Concepts

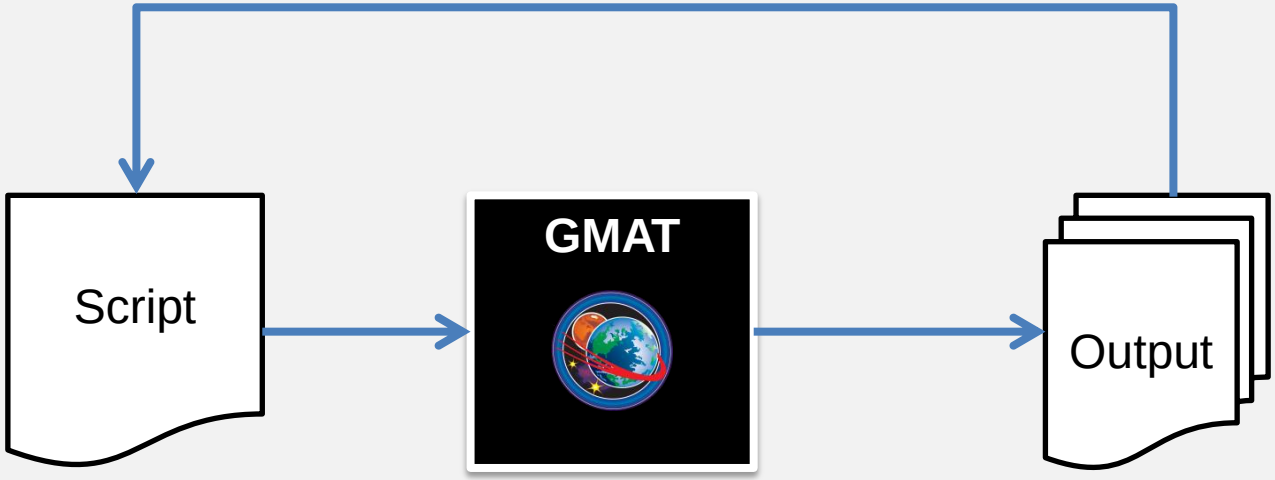
- Execution model
  Why GMAT is more like MATLAB than Excel.
- Two parallel interfaces
  How do they interact?
- Resources and commands
  What are they? What types exist?
- Fields and parameters
  What's the difference? How do I use them?

KC1: Execution Model

- GMAT is like MATLAB:
  - You write a program (a “mission”), then run it to generate output
  - Similar to the FreeFlyer model
- Not like Excel
  - Cannot generate output or manipulate results without rerunning
  - This is the STK model

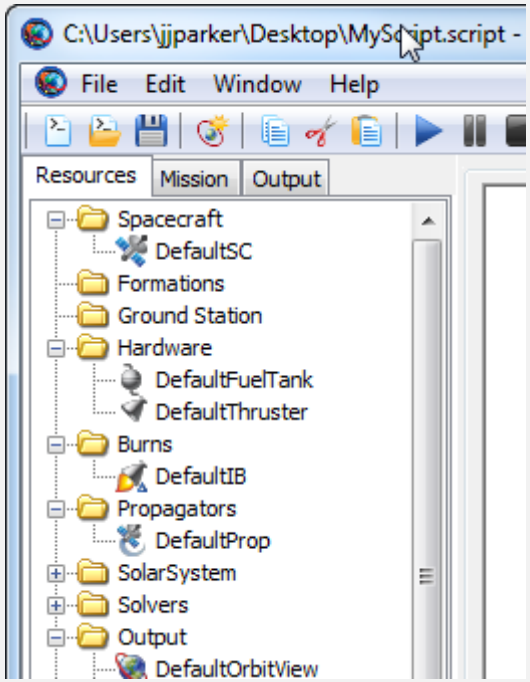
KC1: Execution Model

- Batch execution model



KC2: Two Parallel Interfaces

GUI



Script

```
1  %General Mission Analysis Tool(GMAT) Script
2  %Created: 2013-01-23 09:47:07
3
4
5  %----------------------------------------
6  %---------- Spacecraft
7  %----------------------------------------
8
9  Create Spacecraft DefaultSC;
10 GMAT DefaultSC.DateFormat = TAIModJulian;
11 GMAT DefaultSC.Epoch = '21545';
12 GMAT DefaultSC.CoordinateSystem = EarthMJ2000Eq;
13 GMAT DefaultSC.DisplayStateType = Cartesian;
14 GMAT DefaultSC.X = 7100;
15 GMAT DefaultSC.Y = 0;
16 GMAT DefaultSC.Z = 1300;
17 GMAT DefaultSC.VX = 0;
18 GMAT DefaultSC.VY = 7.35;
19 GMAT DefaultSC.VZ = 1;
20 GMAT DefaultSC.DryMass = 850;
21 GMAT DefaultSC.Cd = 2.2;
```

GUI and script are nearly interchangeable (but not totally).

KC2: Two Parallel Interfaces

GUI-only features

- Interactive features
- This includes:
  - 3D window manipulation
  - Animation
  - 2D plot editing & export
  - Capturing a screenshot
  - Command summary
  - Solver “Apply Corrections”

Script-only features

- Advanced functionality
- This includes:
  - Advanced force models
  - Developmental features
  - Additional forces from plugins
- New development happens in script first, then GUI.

KC3: Resources and Commands

Resources

- Participants in a GMAT mission
- Represent the “things” that will be manipulated
- Think of them as objects, with properties
- Most are “fixed” when the mission starts

Commands

- Events in a GMAT mission
- Represent the actions taken on the resources
- Think of them as methods or functions

KC3: Resources

- **Physical objects**
  - Spacecraft, FuelTank, Thruster, Planet, Asteroid
- **Mission objects**
  - ImpulsiveBurn, FiniteBurn
- **Analysis objects**
  - Propagator, ForceModel, CoordinateSystem, DifferentialCorrector
- **Software objects**
  - Variable, Array, String, EphemerisFile, ReportFile

KC3: Commands

- **Mission events**
  - `Propagate`, `Maneuver`, `BeginFiniteBurn`
- **Analysis statements**
  - `Optimize`, `Target`, `Minimize`, `Vary`
- **Programming statements**
  - `For`, `While`, `CallMatlabFunction`, Assignment (“=”)
- **Output control**
  - `Report`, `Toggle`, `ClearPlot`, `MarkPoint`

KC4: Fields and Parameters

Fields

- Properties you can set on a resource
- Examples:
  - `Spacecraft.Epoch`
  - `Thruster.DecrementMass`
  - `ReportFile.Filename`

Parameters

- Properties you can calculate during the mission
- Parameters often have dependencies
- Examples:
  - `Spacecraft.Earth.Altitude`
  - `Spacecraft.EarthMJ2000Eq.BVectorAngle`

- Sometimes a property is both a field and a parameter.
- Examples: `Spacecraft.SMA`, `FuelTank.FuelMass`

Example: LCROSS

- Mission: Impactor to discover presence of water in permanently-shadowed craters at lunar south pole. Launched with LRO.

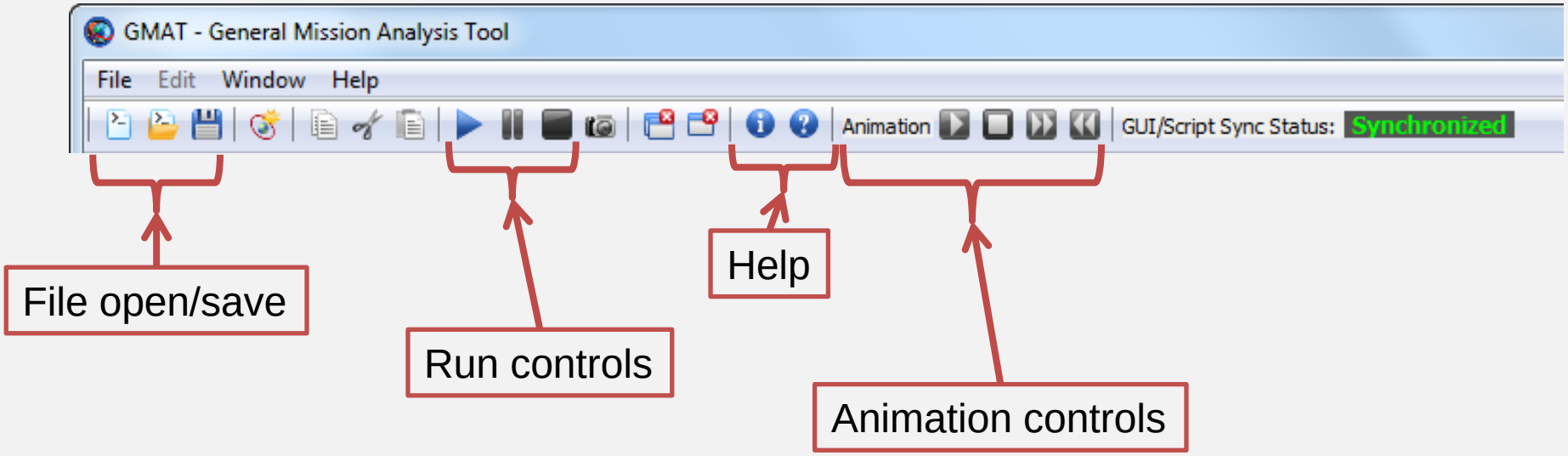
- Example Goal: Optimal lunar flyby to perform phasing and plane change resulting in polar lunar impact.

- `Ex_LCROSSTrajectory.script` shows recreation of optimal trajectory.
- Highly non-linear regime
- Uses segment-based analysis to mitigate nonlinearity:
  - Segment 1: TOI → 120 days (forward)
  - Segment 2: Lunar crater impact → endpoint matching with Segment 1 (backward)
- Optimization to minimize fuel while eliminating segment boundary mismatch.
- Launch window analysis using GMAT and Copernicus software.

TOUR OF THE GRAPHICAL USER INTERFACE

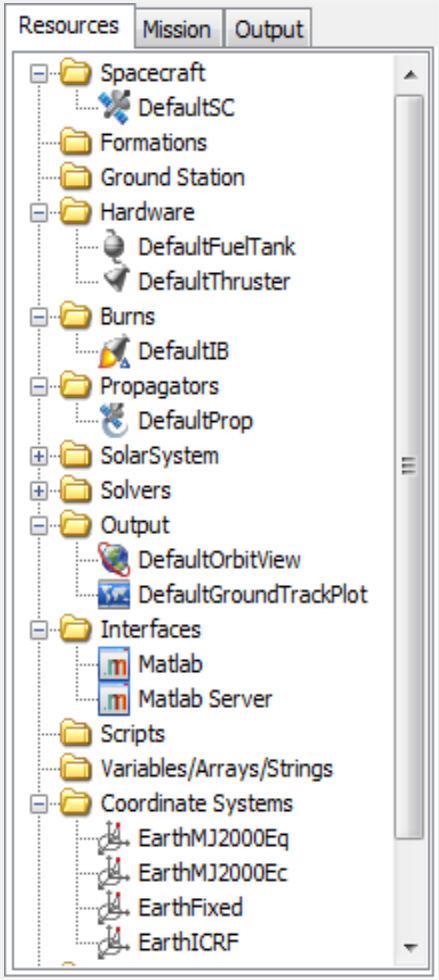
GMAT

Toolbar and Menu Bar



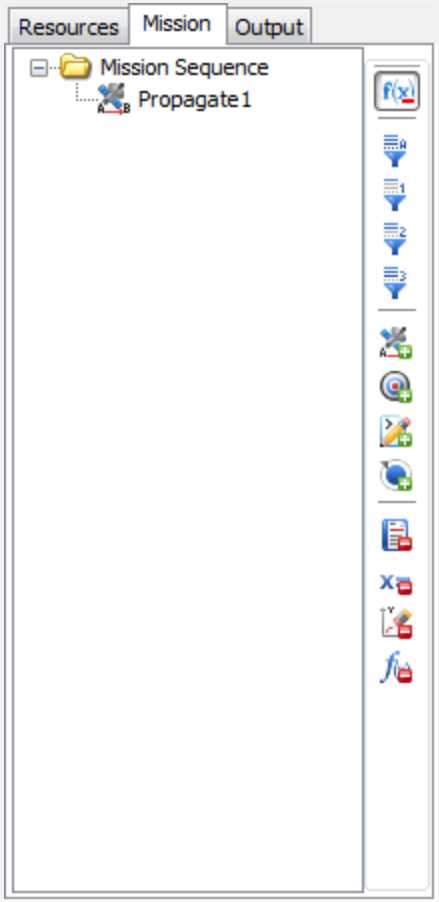
Resource Tree

- Contains all configured resources in the mission
- Grouped into folders by type:
  - Spacecraft
  - Hardware
  - Burns
  - Output
  - SolarSystem



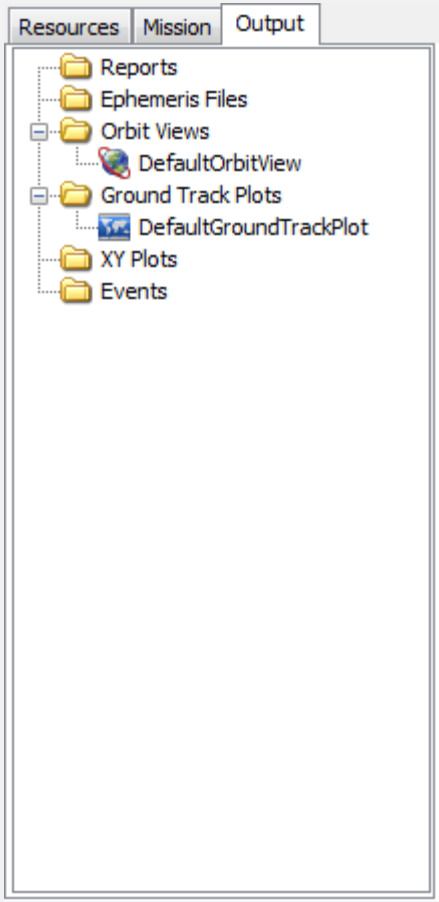
Mission Tree

- Contains the Mission Sequence—sequence of all configured commands
- Special features:
  - Docking & undocking
  - Filtering controls
  - Command Summary



Output Tree

- Contains all output products
- Populated *after* mission execution

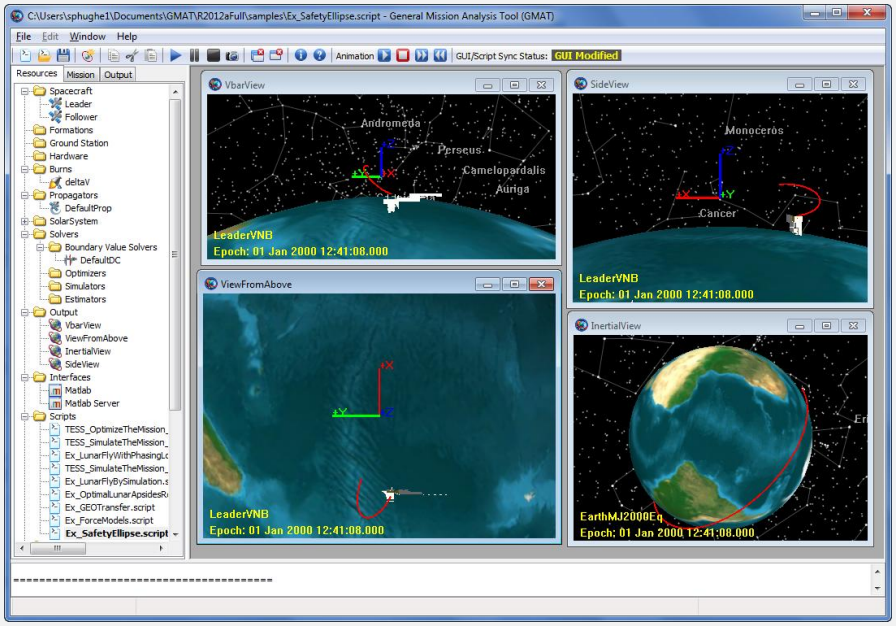


OrbitView

- 3D graphics window
- Most complex of the graphical output types
  - Others include: XYPlot (2D plotting), GroundTrackPlot (2D mapping)
- Mouse controls:
  - Left button: rotation
  - Right button: zoom (horizontal motion)
  - Middle button: rotation normal to screen
- Configuration includes:
  - Camera controls
  - Resources to draw
  - Visual elements

Ex: Safety Ellipse Proximity Operations

- Objectives:
  Simulate proximity operations
- Script:
  - `Ex_SafetyEllipse`
- Highlighted features:
  - Coordinate Systems
  - Graphics



TOUR OF THE SCRIPT LANGUAGE

GMAT

Basic Syntax

- Syntax is based on MATLAB
- Single-line statements w/ optional line continuations
- Case sensitive
- Loosely typed
- Begin/End block statements
- Resources are created before used (except special defaults like SolarSystem)

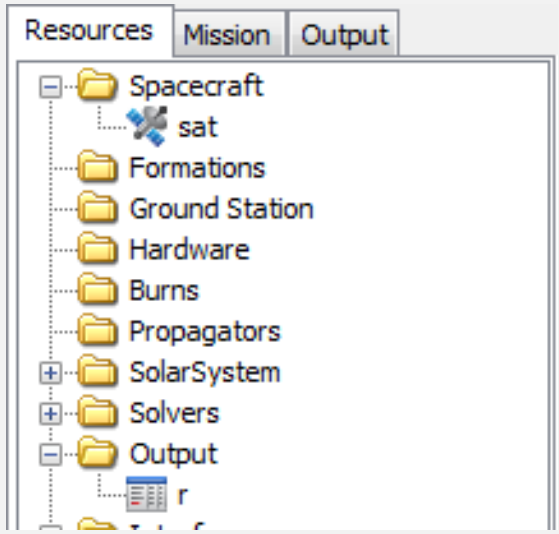
Basic Syntax

- Script is divided into two sections:
  - Initialization (at the top)
  - Mission Sequence (at the bottom)
  - Divided by the `BeginMissionSequence` command
- Initialization -> Resources Tree
  - Static assignment only
- Mission Sequence -> Mission Tree
  - Manipulation of existing resources, cannot create new ones

Basic Syntax

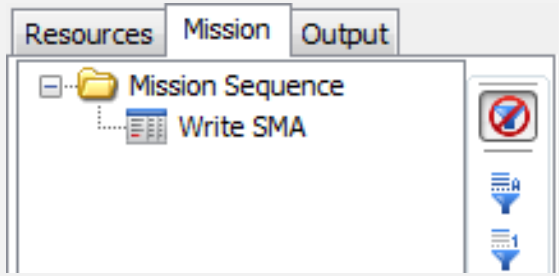
```
Create Spacecraft sat
sat.SMA = 7000

Create ReportFile r
r.Filename = 'MyReport.txt'
```




```
BeginMissionSequence

Report 'Write SMA' r sat.SMA
```



Using Math

- Math syntax is based on MATLAB
- Operators are matrix-aware

Operators	
+	add
-	subtract
*	multiply
/	divide
'	transpose
^	power

Built-in Functions	
sin	cos
tan	asin
acos	atan
atan2	log
log10	exp
DegToRad	RadToDeg
abs	sqrt
norm	det
inv	

Using Math

- Math is allowed only in the context of assignment statements (“=”) only
- Only in the Mission Sequence
  - Corresponds to Equation command in GUI
- This is allowed: `myVar = 24 * 60^2`
- This isn’t: `While x <= y-z`

Using Math

```
Create Spacecraft SC
SC.SMA = 7100
Create Variable period, mu, pi
mu = 398600.4415

BeginMissionSequence

pi = acos(-1)
period = 2 * pi * sqrt(SC.SMA^3/mu)
```

Using Parameters

- Parameters can have one of two types of dependencies (or neither):
  - Central body
  - Coordinate system
- They are calculated on the fly when they are used:
  - `Spacecraft.MarsFixed.X`
  - `Spacecraft.Earth.BetaAngle`
- If omitted, default dependency is used

Using Parameters

```
Create Spacecraft SC
SC.CoordinateSystem = MarsFixed
Create ReportFile r
BeginMissionSequence

% using parameters
Report r SC.X
Report r SC.BetaAngle
```

Using Parameters

```
Create Spacecraft SC
SC.CoordinateSystem = MarsFixed
Create ReportFile r
BeginMissionSequence

% using parameters
Report r SC.EarthMJ2000Eq.X
Report r SC.Earth.BetaAngle
```

Using Parameters

```
Create Spacecraft SC
SC.CoordinateSystem = MarsFixed
Create ReportFile r
BeginMissionSequence

% setting fields
SC.RAAN = 90
SC.ECC = 0.2
```

Using Parameters

```
Create Spacecraft SC
SC.CoordinateSystem = MarsFixed
Create ReportFile r
BeginMissionSequence

% setting fields
SC.MarsFixed.RAAN = 90
SC.Mars.ECC = 0.2
```

Using Parameters

```
Create Spacecraft SC
SC.CoordinateSystem = MarsFixed
Create ReportFile r
BeginMissionSequence

% set field using parameter
SC.ECC = 2 * SC.ECC
```

Using Parameters

```
Create Spacecraft SC
SC.CoordinateSystem = MarsFixed
Create ReportFile r
BeginMissionSequence

% set field using parameter
SC.Mars.ECC = 2 * SC.Earth.ECC
```

DANGER! Always state your dependency.

Control Flow

- Three control flow statements:
  - If/Else – execute if a conditional is true
  - While – loop while a condition is true
  - For – loop a certain number of times

If/Else

```
If SC.Earth.Altitude < 300
    % do a maneuver
Else
    % continue
EndIf
```

While

```
While SC.Tank1.FuelMass > 5
    % continue burning
EndWhile
```

For

```
For i = 1:1:50
    % do something 50 times
EndFor
```

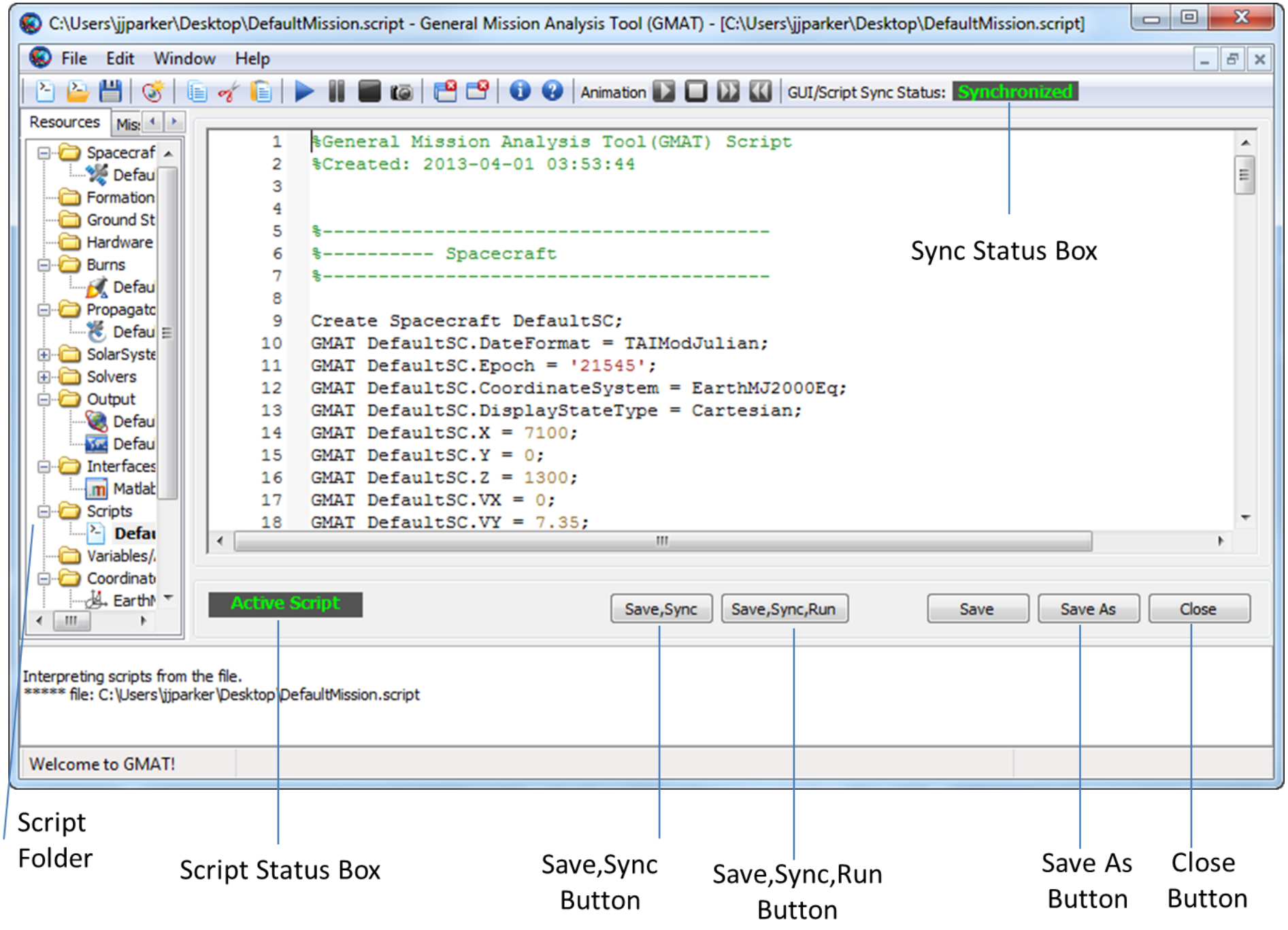
Solvers

- Two types of solvers:
  - `Target` (using `DifferentialCorrector`)
  - `Optimize` (using either optimizer)
- Similar to loops, with specific nested commands:
  - `Target`: `Vary`, `Achieve`
  - `Optimize`: `Vary`, `NonlinearConstraint`, `Minimize`
- See the tutorials for examples

Script Editor

- Built-in editing control
- Includes code folding, syntax highlighting
- GUI/script synchronization indicators:

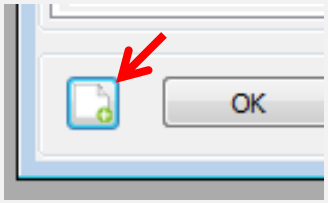
Status	Meaning
Synchronized	GUI and script are identical
GUI Modified Script Modified	GUI or script has modifications
Unsynchronized	GUI AND script have modifications – fix manually
Script Error	Script couldn't be loaded


C:\Users\jjparker\Desktop\DefaultMission.script - General Mission Analysis Tool (GMAT) - [C:\Users\jjparker\Desktop\DefaultMission.script]
File Edit Window Help
Animation
GUI/Script Sync Status: Synchronized
Resources
Spacecraf
Defau
Formation
Ground St
Hardware
Burns
Defau
Propagatc
Defau
SolarSyste
Solvers
Output
Defau
Defau
Interfaces
Matlab
Scripts
Defau
Variables/
Coordinat
EarthM
1 %General Mission Analysis Tool(GMAT) Script
2 %Created: 2013-04-01 03:53:44
3
4
5 %--
6 %---------- Spacecraft
7 %--
8
9 Create Spacecraft DefaultSC;
10 GMAT DefaultSC.DateFormat = TAIModJulian;
11 GMAT DefaultSC.Epoch = '21545';
12 GMAT DefaultSC.CoordinateSystem = EarthMJ2000Eq;
13 GMAT DefaultSC.DisplayStateType = Cartesian;
14 GMAT DefaultSC.X = 7100;
15 GMAT DefaultSC.Y = 0;
16 GMAT DefaultSC.Z = 1300;
17 GMAT DefaultSC.VX = 0;
18 GMAT DefaultSC.VY = 7.35;
Sync Status Box
Active Script
Save,Sync
Save,Sync,Run
Save
Save As
Close
Interpreting scripts from the file.
***** file: C:\Users\jjparker\Desktop\DefaultMission.script
Welcome to GMAT!
Script Folder
Script Status Box
Save,Sync Button
Save,Sync,Run Button
Save As Button
Close Button

Best Practices

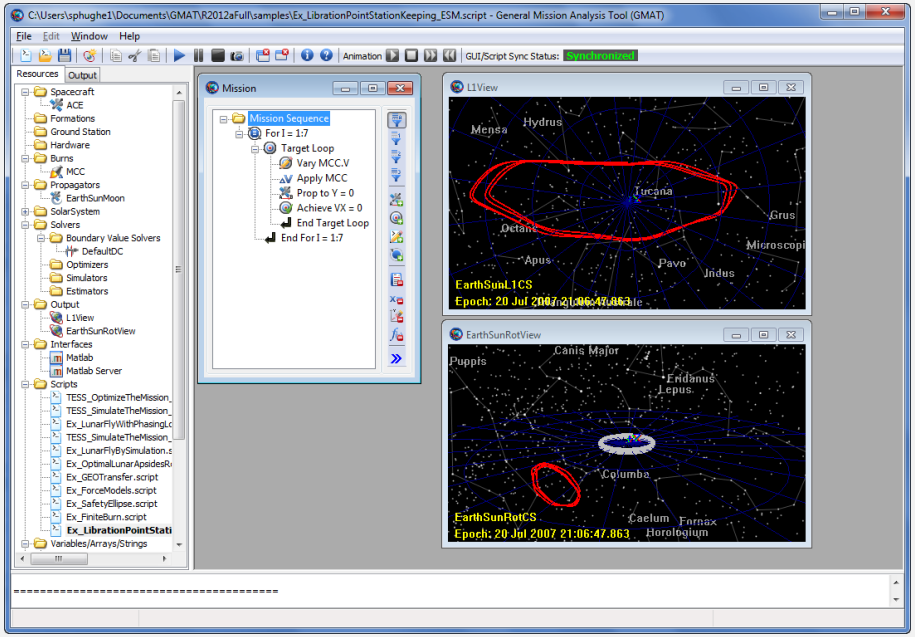
1. Keep script minimal
   - Missing field settings will remain at default values
2. Use the “Show Script” button on each GUI panel
3. Always explicitly state parameter dependencies
   - (Do as I say, not as I do—examples compressed.)
4. Use `ScriptEvent` to encapsulate complex algorithms
5. Label your commands:

```
Report 'Write SMA' r SC.SMA
```

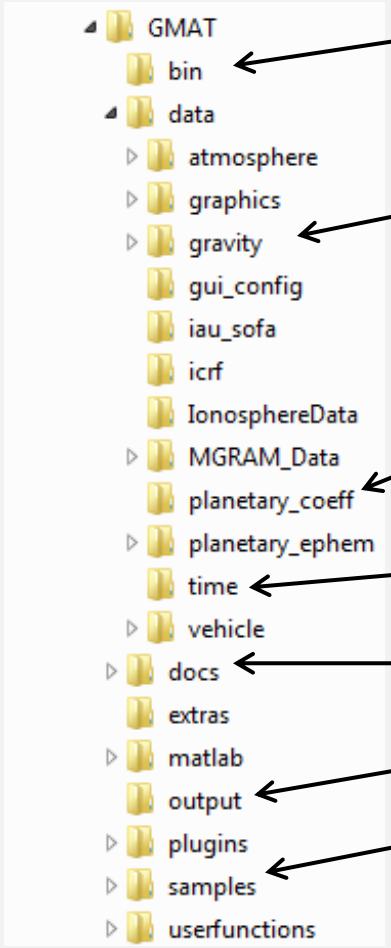


Ex: Libration Point Station-Keeping

- Objective: Maintain a libration point orbit
- Script
  - `Ex_LibrationPointStationKeeping_ESM.script`
- Highlighted Features
  - Coordinate Systems
  - Graphics
  - Control Flow
  - Targeting



Data Files and Configuration



configuration (`gmat_startup_file.txt`)

gravity models

EOP file

leap second files

documentation (math specs, user guide)

output files

samples & tutorial scripts

Plugins

- GMAT ships with several default plugins.
- Two require user configuration:

Plugin	Description
`libFminconOptimizer`	MATLAB fmincon optimizer interface
`libMatlabInterface`	Run MATLAB functions

- Enable or disable them through `PLUGIN` lines in `bin\gmat_startup_file.txt`
- Also, several alpha-level plugins are available. Ask for details.
  - C interface, estimation, script functions, alpha integrators, Mars-GRAM 2005 density model, object serialization

Getting Help

- For feature-specific information:
  - Help button on feature panel
- For scripting help:
  - “Show Script” button on feature panel
- Overall information:
  - GMAT User Guide (Help > Contents)
  - Updated copy: http://gmat.sf.net/docs/nightly

Getting Help

- GMAT Wiki:
  - http://gmatcentral.org/
- User Forum
  - http://forums.gmatcentral.org/
- Mailing lists:
  - gmat-users@lists.sourceforge.net
  - gmat-developers@lists.sourceforge.net
  - Subscribe at http://sf.net/projects/gmat