Integrating & Customizing STK with Plugin Scripts

Required License: STK Pro
Estimated Completion Time: 90 min

What you will do

- Load a Vector Geometry Tool plugin and create a graph on the resulting vector.
- Activate an access constraint plugin, set the desired threshold, and graph the access times.

What you will learn

The following activity introduces you to the STK plugin scripts. You will learn about the basic plugin structure, how inputs and outputs are passed to STK, and the pros and cons of each scripting language. You will look at different plugin types and identify the common and custom functions.

Using the Plugins, you will solve a couple of problems. First, you will use the Vector Geometry Tool (VGT) and apply a plugin to compute the angle of a mirror so that sunlight reflects from a heliograph station to another as a method of communication. Then, you will use an Access Plugin to compute when access will occur between two ground stations via the heliograph.

Tutorial scenario

You can find a complete STK scenario for this tutorial at:

<STK install folder>\Data\Resources\stktraining\samples\HeliographComplete\...

About plugin scripts in STK

Plugin scripts enable customization of STK when built-in models may be inadequate. You can create and incorporate new functions without changing STK itself. These scripts talk with STK via entry points that are designed into the application. You can write the scripts in VBScript, MATLAB, or JavaScript. The benefit compared to previous STK customization methods is that you need not understand how STK is coded.

Every entry point has common and unique data protocols that are documented in the STK Programming Interface help. The rules differ slightly from one entry point to another; however, the scripting methodologies are nevertheless closely related. The five main plugin types are summarized below:

STK Plugin Types	Example Applications
Astrogator 1. Calculation Objects 2. Custom Engine Models 3. Propagators 4. Vectors	• Custom engine thrust, mass flow rate, I_{SP} • External forces in addition to or different from those in STK
Vector Geometry Tool 1. Custom Vectors 2. Custom Axes 3. Custom Calculations	• Time-varying vector or axes • Vectors and axes based on selection strategies • Custom equations in a single calculation component
Attitude Simulator	• External torques • Attitude control laws
Access Constraint	• Customized visibility constraints
Communication 1. Transmitter Model 2. Receiver Model 3. Antenna Gain Model 4. Rain Model 5. Gaseous Absorption Model 6. Multi-beam Antenna Model 7. Satellite Selection Model 8. Comm Constraint Model	• Custom propagation losses and absorptions • Custom dynamic antenna gains • Custom antenna selection strategies

Exercise

You will use a Vector Geometry Tool (VGT) plugin with the Analysis Workbench to compute the angle of a mirror so that sunlight reflects from a heliograph station to another as a method of communication. Begin by setting up some basic things in your scenario.

1. Open the Heliograph scenario from the following location:

   <STK install folder>\Data\Resources\stktraining\samples\Heliograph.

   You should have two facilities: one on White Mountain and another on Sherwin Peak. Set your view position to White_Mountain. Animate the scenario. You should see three vectors, one

pointing at the Sun, one pointing at Sherwin Peak, and one pointing north. The bisection of the Sun and station vectors is the target for the new vector.

Next, you will load a plugin that defines the mirror vector based on the vectors of Sherwin Peak and the Sun. To use a plugin, you must place the plugin in one of a few directories. Double-check that your scenario folder has a Scripting folder. If not, you can find the folder and contents at <STK install folder>\Data\Resources\ stktraining\samples\\Heliograph\Scripting, in the scenario directory. The complete path for the plugin you'll be using in the first section is below:

… <Scenario Directory>\Scripting\VectorTool\Vector\HelioVector.vbs

You will analyze the file in detail later.

Note: There is also a HelioVector.m for a Maltab version of the plugin script. You will have to download and install the STK Matlab Connectors for the script to work. You can read more about the Connectors in the help system under How to Install STK > STK Installation Guide > Using a MATLAB Connector

2. Open the VGT panel for White_Mountain by highlighting the facility, right-clicking, and selecting Analysis Workbench.
3. Click the Create new Vector button to create a new vector. The Add Geometry Component page will appear.
4. Enter the vector Name as "Mirror" and set Type to Custom Script.
5. Under Vector Script File, click Select … and highlight HelioVector.vbs. Click Close.
6. Close out the VGT page by clicking OK and then Close.

   The plugin is now loaded.

7. Bring up the Message Viewer.

   Ensure the last line says CustomVector script HelioVector.vbs initialized and working. This indicates the registration of inputs and outputs was successful.

   Next, show the Mirror vector in the 3D Graphics window.

8. Open the Properties for White_Mountain.
9. Go to the 3D Graphics-Vector page.
10. Select the Vectors tab and click Add…, highlight Mirror in the right column, and click Apply and then Close.
11. Select the Show Label check box and set Scale Relative to Model to 2.0.
12. Press OK.

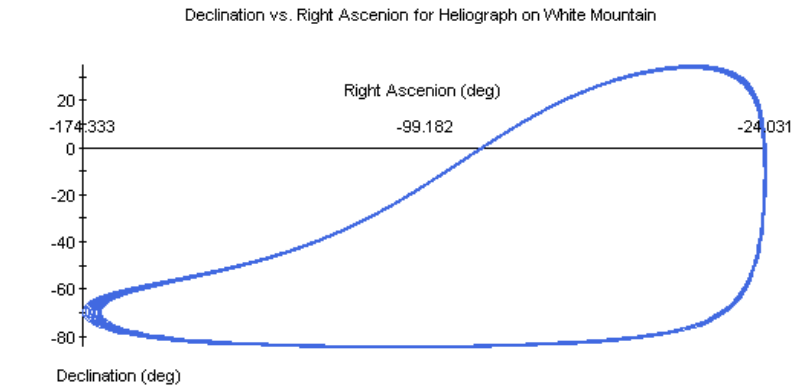
    You will see a VBScript pop-up that is triggered the first time the script runs; this feature was written in the plugin and is not part of STK. Click Ok to dismiss it.

    Animate the scenario. The new vector should be visible at White Mountain and should bisect the station and Sun vectors.

    Next, look at the analytical information for the Heliograph (Mirror Vector). You will create a graph of the data.

13. Highlight White_Mountain in the object browser.

14. Right-click White_Mountain and open the Report & Graph Manager.
15. Under Heliograph Styles>Styles>Facility, select Mirror_Graph; this is a custom graph style located in the scenario folder.
16. Click Create. You can select Properties … to see what data providers were used to construct the style.



Time of nearest point is 1 Jun 2004 13:12:00.00

This graph shows the position for the Heliograph on White Mountain, a direct result of computations in our plugin.

17. Save your scenario.

Plugin Structure

Plugins communicate with STK via built-in entry points that expect a certain input and output protocol. One array for input and one array for output are exchanged between STK and the plugin for every instance the script is called.

Plugins are activated in two possible ways. Some plugins, like access constraints, must reside in a specific folder before STK startup and they stay activated during the STK session. Other plugins are activated by the user pointing to the plugin file in the GUI and can be turned on and off within a session.

Once activated, STK relies on the plugin for specific computations. Depending on the plugin type, they may get solicited at different times, such as during an integration time step or a simulation time step.

Before the custom function gets executed for the first time, STK performs a registration of arguments that later get passed to and from STK. This registration call mode is different from the computation call mode or other call modes, and to keep them apart a gateway function is used.

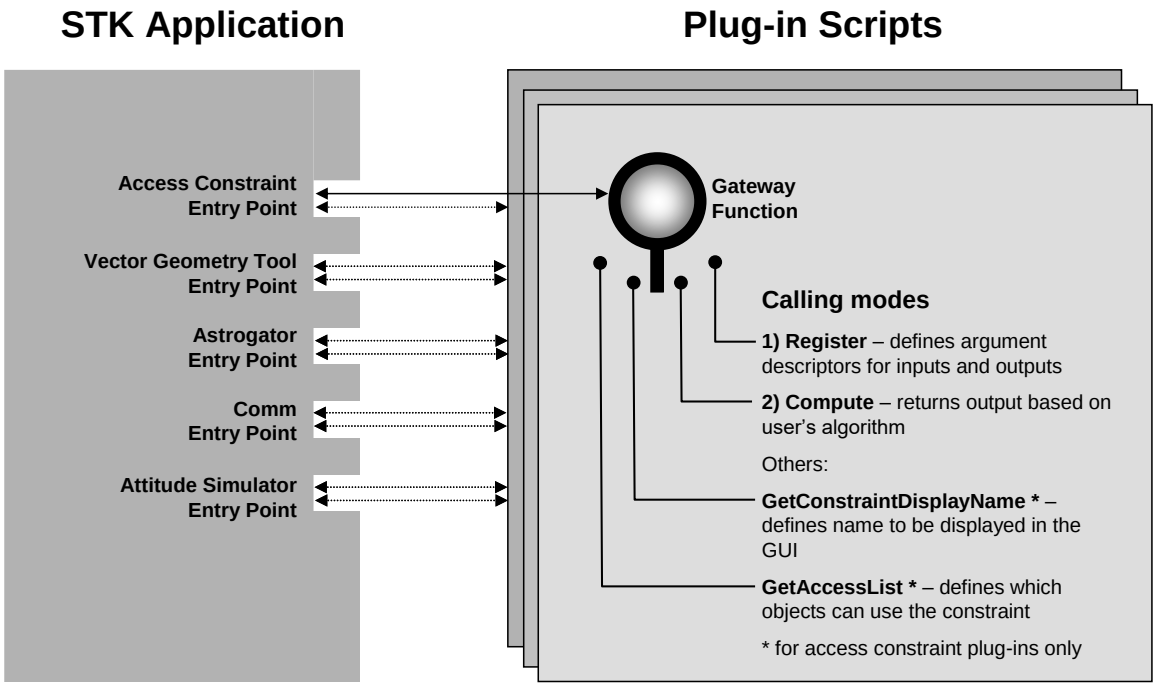


Figure 1 STK plugin script diagram

Plugin argument registration

Before the plugin can run computations, STK needs to identify the arguments that are supplied and received. It first calls the plugin with a 'register' call mode. The gateway function channels the call to the register function where these arguments are defined.

Sample definition from "STK Plugin Scripts" Reference

Keyword	Value
Type	Vector
Name	Name of the requested vector
Source (optional)	`STK path for the object where the vector of given Name resides`
RefName	Name of requested reference axes
RefSource (optional)	STK path for the object where the axes of given RefName reside

Sample registration in VBScript based on the table above

```
descripStr(0) = "ArgumentType = Input"
descripStr(1) = "Type = Vector"
descripStr(2) = "Name = Sun"
descripStr(3) = "Source = Facility/White_Mountain"
descripStr(4) = "RefName = Body"
descripStr(5) = "RefSource = Facility/White_Mountain"
```

Attitude and Plugins

You will load a 3D model to better illustrate the motion of the newly created vector.

1. Go to Insert/From File…
2. Select Mirror.gv from your scenario folder <STK install folder>\Data\Resources\stktraining\samples\Heliograph and press Open.

   NOTE: The Mirror ground vehicle is a "dummy" object for display purposes. It is only a device for displaying models that can assume an arbitrary attitude. Keep in mind our custom vectors and constraints are still tied to the White Mountain facility.
3. Open the Properties for Mirror and verify steps 4-7 are complete.
4. Go to the Basic - Attitude page and set the Type to Aligned and Constrained.
5. For the Aligned Vector, set X to 1, and Y and Z to 0. For its reference vector, press Select … and choose Mirror under White_Mountain.
6. Click Close.
7. For the Constrained Vector, set Z to 1, and Y and X to 0. For its reference vector, press Select … and choose East under Mirror.
8. Click Close and then OK.

   Animate the scenario. The mirror model should now track the mirror vector. Look at White Mountain along the Sherwin Peak vector. Does it appear to reflect sunlight towards Sherwin Peak?
9. Save your scenario.

Plugin computation

Once arguments have been registered, the call mode switches to "compute" and the gateway function channels the call to the user defined algorithm. The first task the algorithm does is transfer the registered variables to local variables for subsequent manipulation.

This transfer can occur via name or via array index number. In MATLAB the transfer occurs via name automatically. In VBScript, the simplest approach is to transfer the variables via array index number in the order that the variables were initially registered. If you want to transfer via name, you will need separate helper functions (STK automatically finds these functions in the folder <STK install folder>\STKData\Scripting\Init\...).

To see an example, HelioVector.vbs uses the array index method, and HelioVector2.vbs uses the name method.

Sample value transfer in VBScript via array index

```
Function HelioVector_compute(stateData)
        Dim MyVector1, MyVector2
        MyVector1 = stateData(1)
        MyVector2 = stateData(2)
```

Sample value transfer in VBScript via name

```
Function HelioVector_compute(stateData)
        'this function allow data to be called by name rather than array index
        Set HelioVector_Inputs = g_GetPluginArrayInterface("HelioVector_Inputs")

        Dim MyVector1, MyVector2
        MyVector1 = stateData(HelioVector_Inputs.MyVec1)
        MyVector2 = stateData(HelioVector_Inputs.MyVec2)
```

Comparison of Plugin Languages

VBScript

- Freely available with Microsoft Windows
- Interfaces well with other Microsoft COM objects (e.g. MS Office)
- Fast execution
- Limited Math functions. For example, it does not support $\sin^{-1}$ or $\cos^{-1}$ functions. The workaround is to use expressions involving $\tan^{-1}$ which is supported.

MATLAB

- Can be purchased from www.mathworks.com
- Interpreted execution is slower. Compiled MATLAB execution is faster however.
- Excellent math function libraries and good availability to 3rd party libraries
- Supports matrix/vector operations directly
- Naming by argument name inherently supported
- Easier plugin implementation, easier gateway function, easier plugin registration.
- Visit www.agi.com/matlab for more information and to download connectors
- A sample Matlab vector plugin “HelioVector.m“ is available in the following directory:

<STK install folder>\Data\Resources\
stktraining\samples\Heliograph\Scripting\VectorTool\Vector

Access constraint plugin exercise

You will create access between White Mountain and Sherwin Peak to display times when communication via heliograph is possible.

You want to restrict access from White Mountain to sunlit conditions.

1. Open the Properties for White_Mountain and go to the Constraints - Sun page.
2. Set the minimum sun elevation to 2 deg and click OK.

3. Right-click White_Mountain and select Access.
4. In the page that pops up, select Sherwin Peak and press Compute.
5. Then click Close.

   Animate the scenario and confirm that access only occurs during daylight.

   You want an additional constraint that limits the angle between Sherwin Peak and the Sun. A single mirror heliograph cannot function when the Sun-to-station angle grows beyond ~90 deg or is close to zero.
6. Open Analysis workbench for White_Mountain and create a new Angle.
7. Set Name to HelioAngle and set Type to Between Vectors.
8. Set the From Vector to Sherwin Peak Vector under the White Mountain object.
9. Set the To Vector to Sun Vector also under the White Mountain object.
10. Click OK and Close.
11. Display HelioAngle by opening the properties for White_Mountain and going to the 3D Graphics - Vector page.
12. Click the Angles tab, click Add..., highlight HelioAngle in the right column and click Apply and then Close.
13. Select the Show Label and Show Angle Value check boxes, and verify the Angle Size to be 0.70. Click OK.

    Animate the scenario and verify the angle displays properly.
14. Save your scenario.

    You will now load the access constraint plugin.
15. Verify that the access constraint plugin HelioAccess.vbs is in the following directory:

    ... <Scenario Directory>\Scripting\Constraints\HelioAccess.vbs
16. Copy the Scripting folder.
17. Paste the Scripting folder into My Documents\STK 12\Config or My Documents\STK 12 (x64)\Config depending on what bit-version of STK you are using. If you need to move the plugin into the folder, you will need to close and restart STK.
18. Open the properties to White_Mountain and go to the Constraints – Plugins page.

    Does the plugin show up?
19. Set Min to 2 degrees and Max to 90 degrees (the units are defined in the plugin).
20. Click OK.
21. Recalculate access from White Mountain to Sherwin Peak.

    You will see a VBScript pop-up that is triggered the first time the script runs; this feature was written in the plugin and is not part of STK.

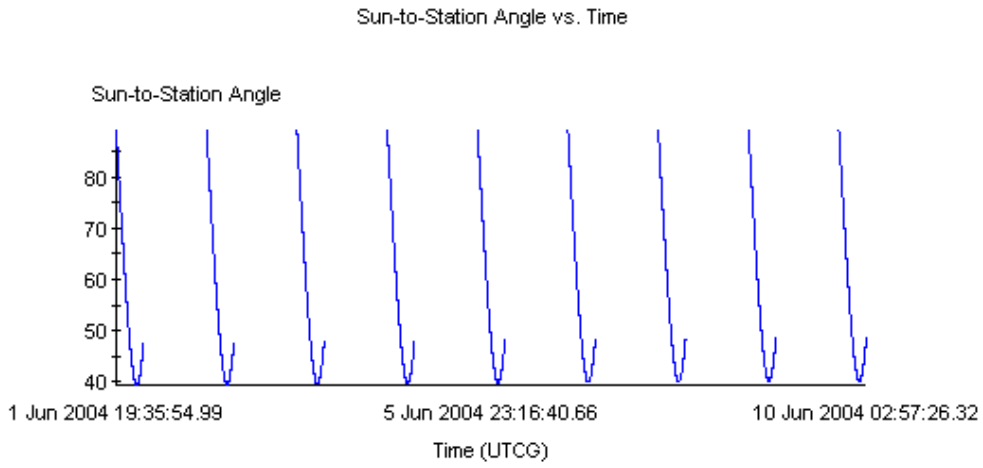
    NOTE: With Connect, you could have assigned the plugin constraint levels by using the plugin name: “SetConstraint */Facility/White_Mountain SunStationAngle Min 2 Max 90”.

22. Look at the message window and ensure the last message says “Custom Access Constraint HelioAccess.vbs initialized and working.” This indicates the registration of inputs and outputs was successful.

Animate the scenario. The new constraints should limit access during sunlight and for Sun-to-Station angles less than 90° degrees. As a side note, you could have constrained HelioAngle via Properties/Constraints/Vector and not used the access plugin at all. The plugin method however can accommodate more variables and handle dynamic definitions – such as the minimum of 2 angles – whereas the built-in constraint capability is more limited.

Next, get the data out by creating a graph of the Sun-to-Station angle for the times when access occurs.

23. Open the Report & Graph Manager.
24. Change the Object Type to Access and highlight the Facility-White_Mountain-To-Facility-Sherwin_Peak
25. Under Heliograph Styles>Styles>Access, select Sun_Station_Angle (this is a custom graph style located in the scenario folder).
26. Ensure Generate As: Report/Graph is enabled and click Generate… You can later select Properties with Sun_Station_Angle highlighted to see what data providers were used to construct the style. Notice the data provider Data Constraints has FromSunStationAngle and ToSunStationAngle that stem from your access constraint plugin.



Analyze the results. Realize this data is a direct result of the plug–in analysis.

Historical Reference

The practice of reflecting sunlight with mirrors as a means for communication is called heliography. The heliograph included a quick release shutter that could be used to transmit Morse code. The British Army first used heliographs during the 1870’s in sunny regions. A 5-inch mirror could communicate up to 70 miles. When the angle between the sun and the target increased above 90 deg a secondary mirror could be used.

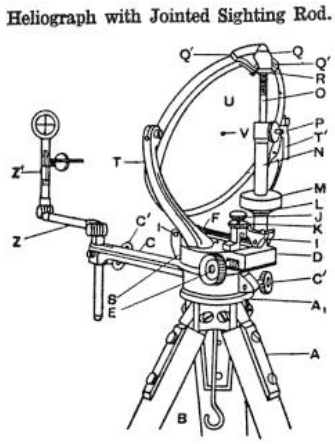


Figure 2 Early model heliograph

Appendix A: Plugin Entry Points

Astrogator

Astrogator plugins are loaded via the Component Browser's Custom Functions (see Figure 3).

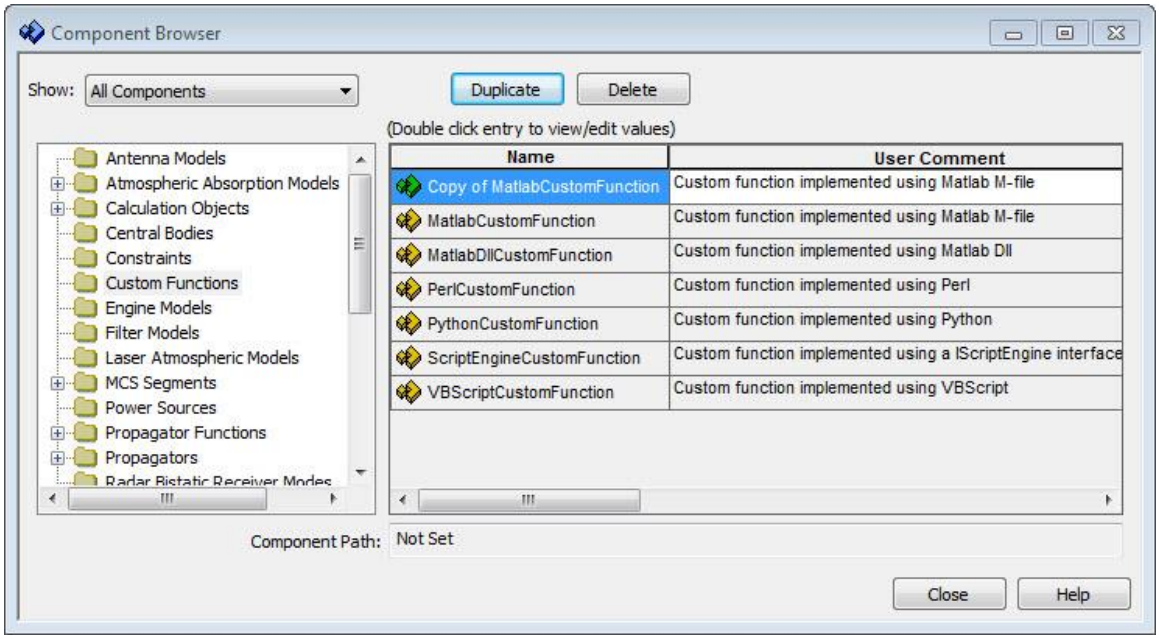


Figure 3 Loading a plugin via the Astrogator Component Browser

Using the Duplicate button, make a copy of the VBScript, MATLAB, or Perl custom function represented in red. Name the copy — now represented in green to denote a user-supplied component — with an appropriate name. Double-click the new component and specify the plugin file location. The new custom function can now be accessed by other Astrogator components.

Required Plugin File Location: No special directory is required. The recommended location is the scenario's folder.

Required License: STK Premuim Space

Vector Geometry Tool

Vector Geometry Tool plugins are loaded when modifying or creating a new vector or axis (see figure 4).

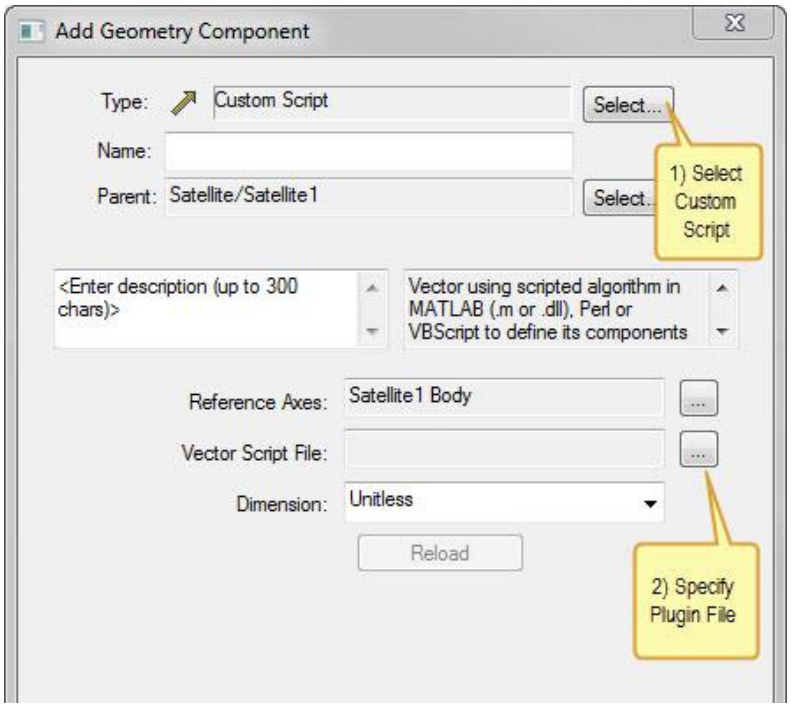


Figure 4 Loading a plugin via Vector Geometry Tool in Analysis Workbench

The vector definition window will pop up when creating or modifying a vector. Select Custom Script under the type, and then click Select… to specify the plugin file location.

Required Plugin File Locations:

Select the level or scope for the plugin: application, user, or scenario.

For Plugin Vectors:

Application level	<STK install folder>\STKData\Scripting\ VectorTool\Vector
User level	< User Directory >\Config\Scripting\VectorTool\Vector
Scenario level	< Scenario Directory >\Scripting\VectorTool\Vector

For Plugin Axes:

Application level	<STK install folder>\STKData\Scripting\VectorTool\Axes
User level	< User Directory >\Config\Scripting\VectorTool\Axes
Scenario level	< Scenario Directory >\Scripting\VectorTool\ Axes

For Plugin Scalars :

Application level	<STK install folder>\STKData\Scripting\VectorTool\Scalar
User level	< User Directory >\Config\Scripting\VectorTool\Scalar
Scenario level	< Scenario Directory >\Scripting\VectorTool\Scalar

Required Module: Analysis Workbench

Attitude Simulator

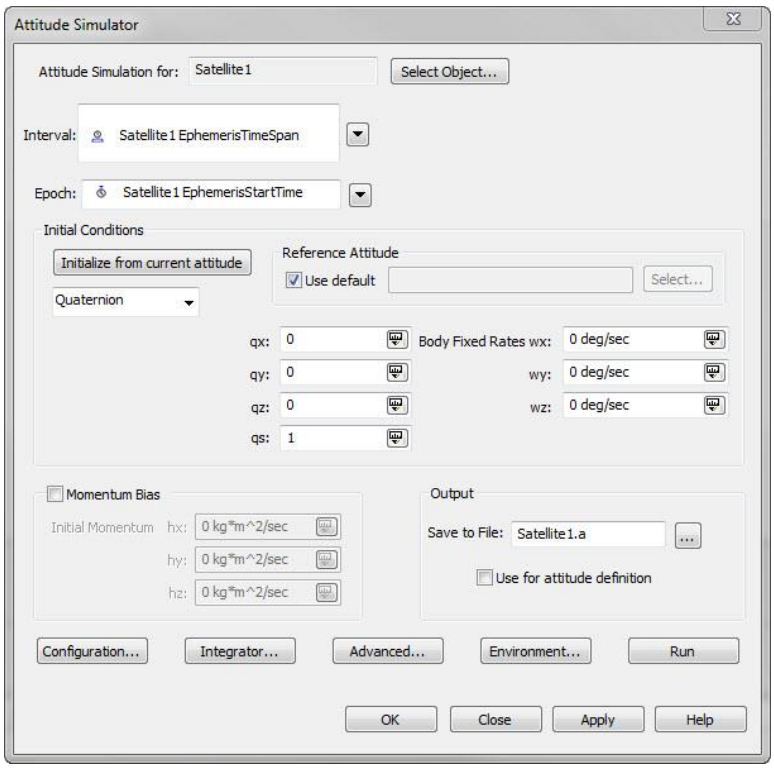


Figure 5 The Attitude Simulator main window

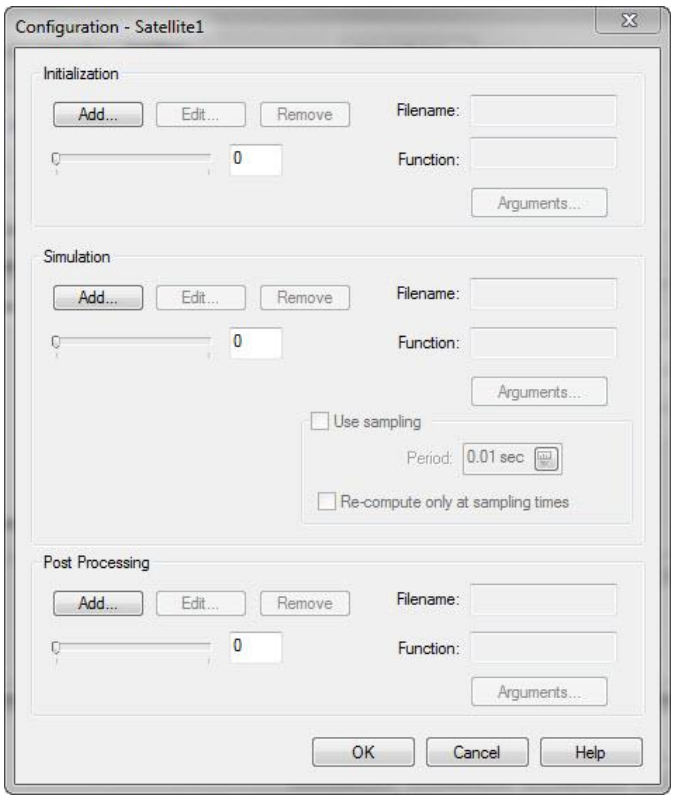


Figure 6 Loading a plugin via the Attitude Configuration panel

Required Plugin File Locations:

App. level	<STK install folder>\STKData\Scripting\Attitude
User level	< User Directory >\Config\Scripting\Attitude
Scenario level	< Scenario Directory >\Scripting\Attitude

Required Module: SatPro

Access Constraints

Access Constraint plugins are automatically loaded when STK starts up. They show up in the constraints properties for all object types that are specified in the plugin.

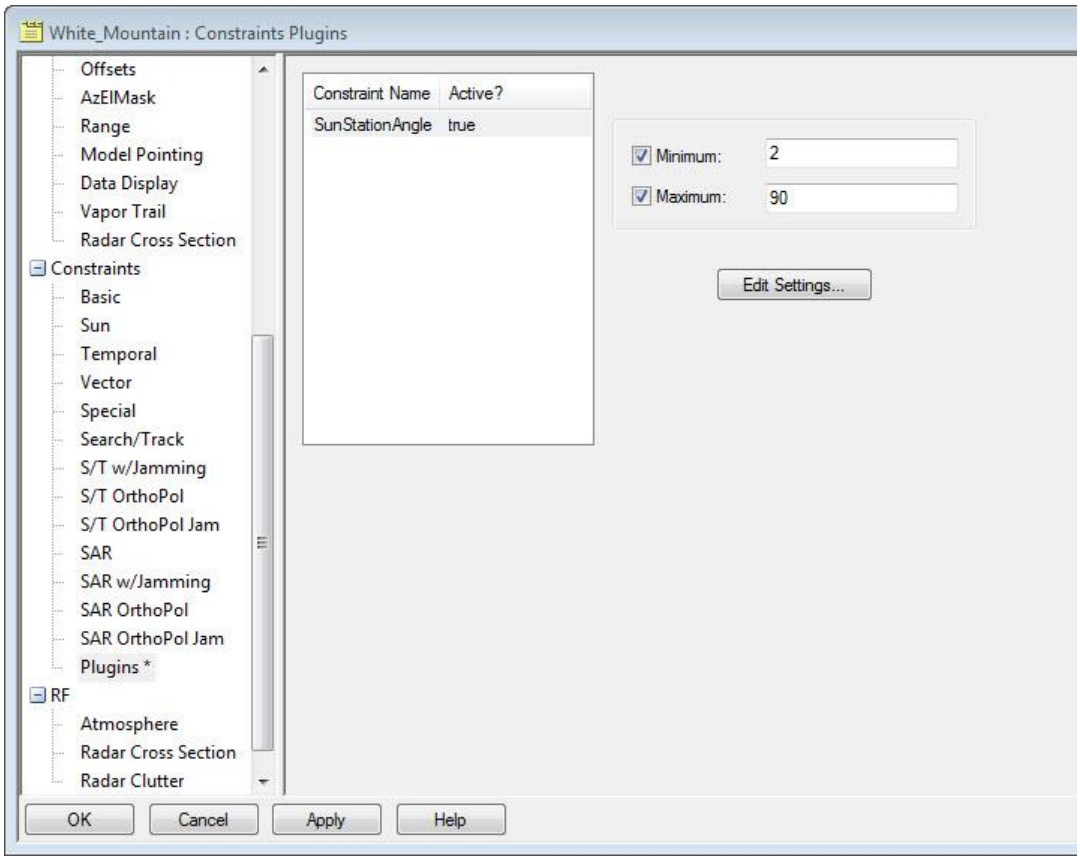


Figure 7 Setting the access constraint plugin

Required Plugin File Locations:

App. level <STK install folder>\STKData\Scripting\Constraints

User level < User Directory >\Config\Scripting\Constraints

Scenario level < Scenario Directory >\Scripting\Constraints

Required Module: STK Pro

Communications

Communications plugins are loaded via several possible settings.

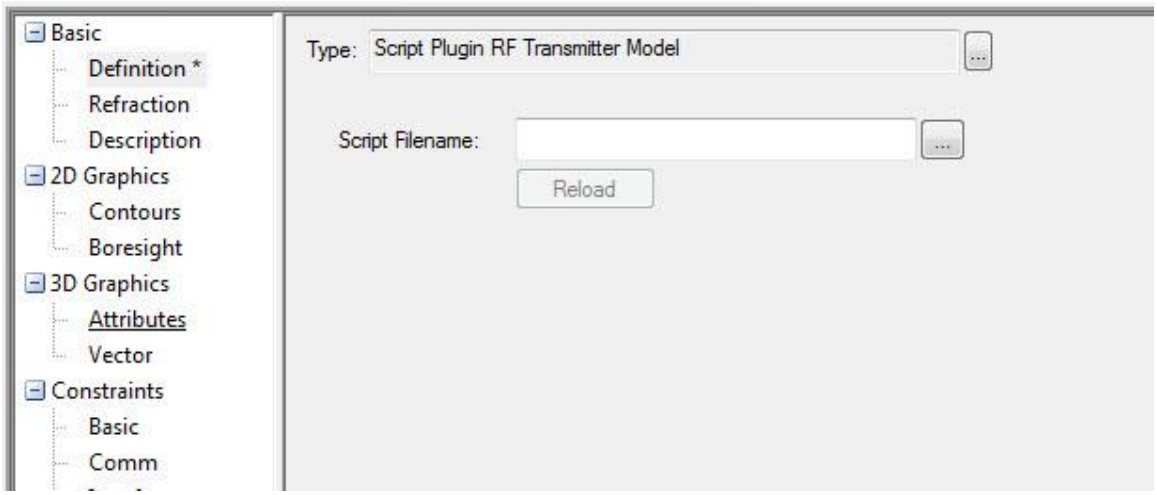


Figure 8 Transmitter model plugin entry point

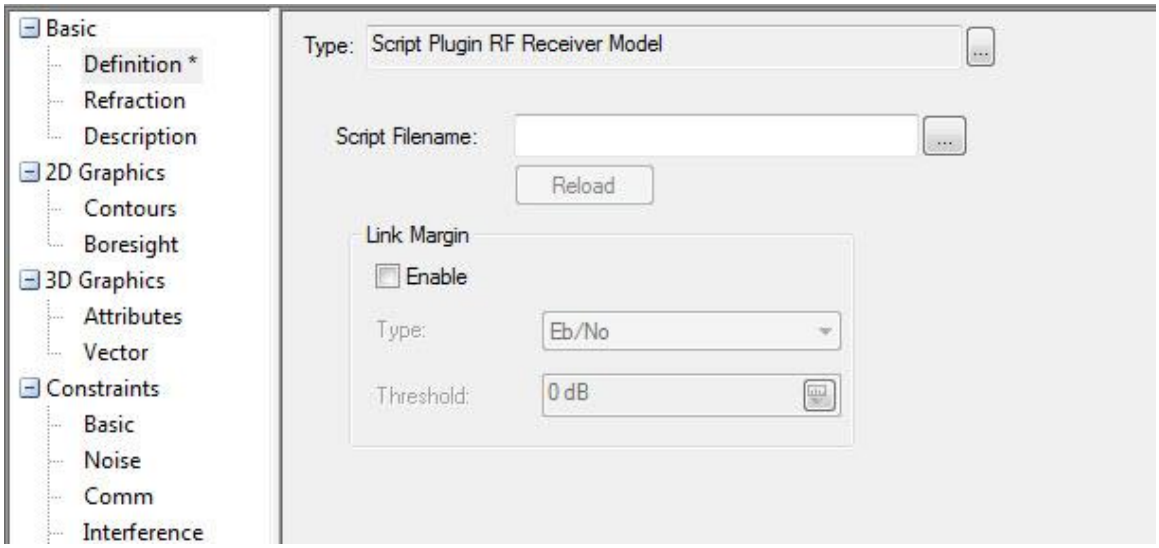


Figure 9 Receiver model plugin entry point

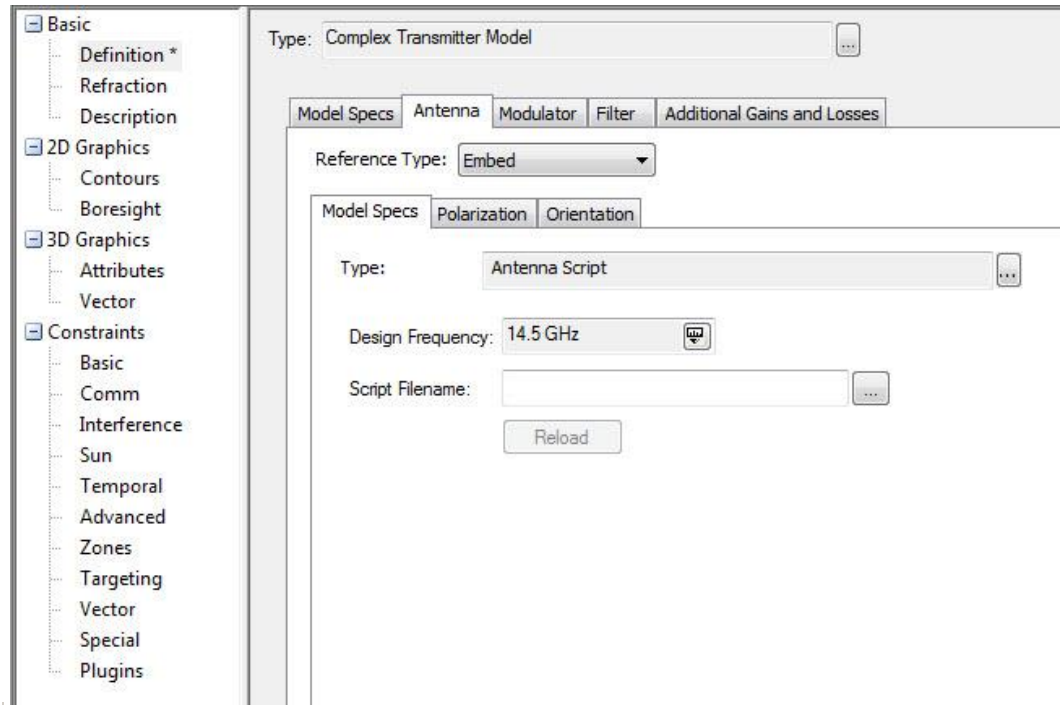
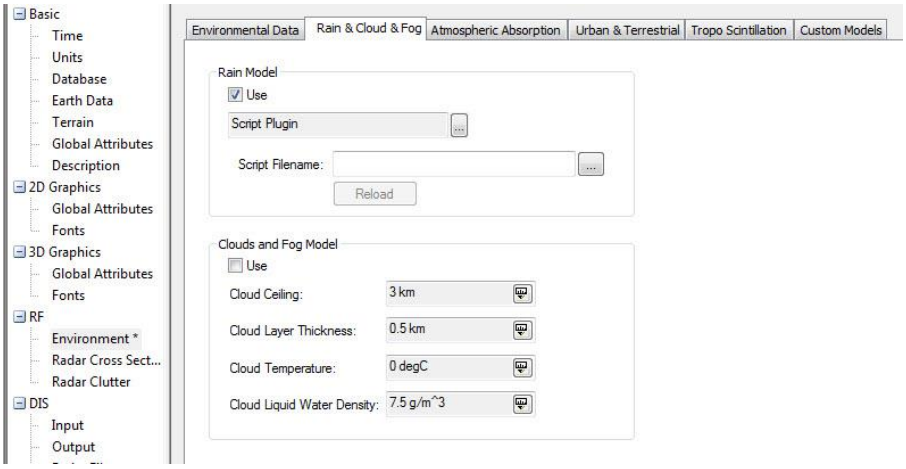


Figure 10 Gaussian Antenna Gain plugin entry point



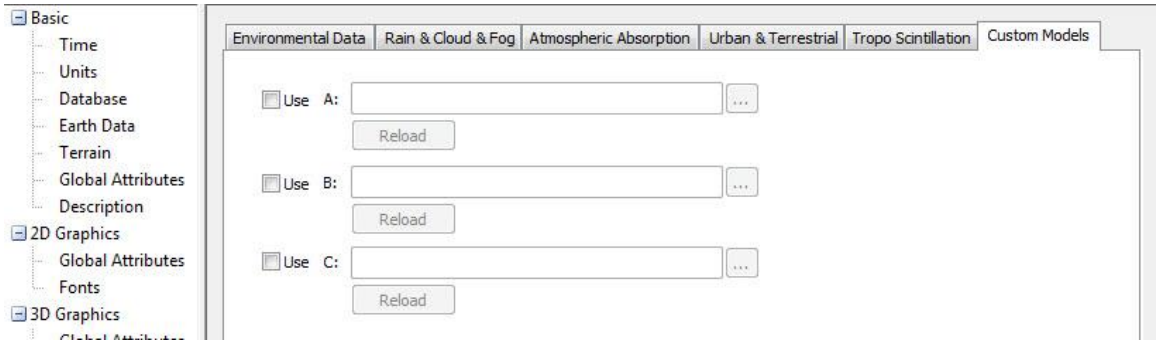


Figure 11 Loading Rain Loss Model and Absorption Model plugin entry points

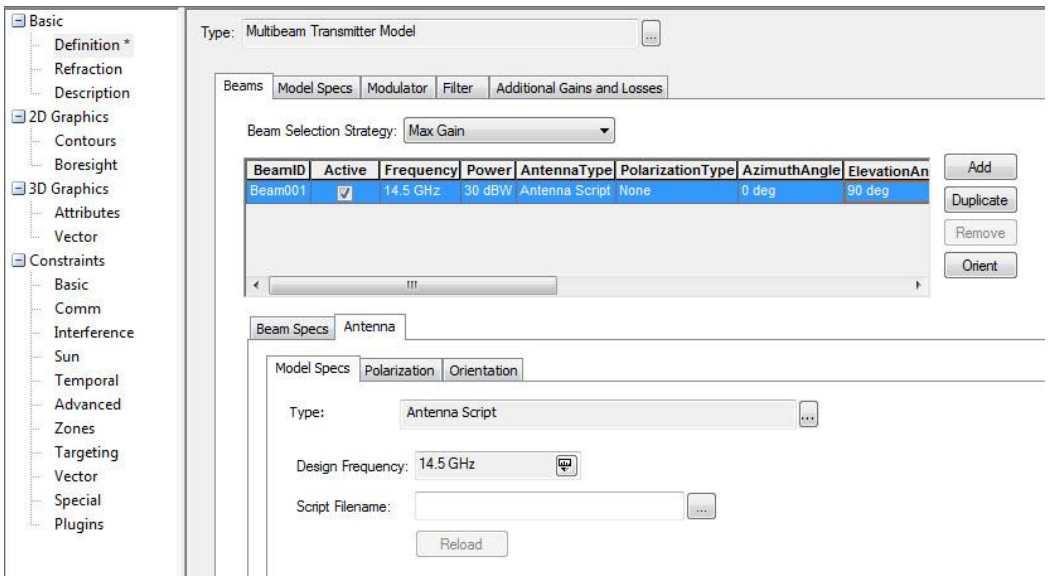


Figure 12 Antenna Multibeam Selection Strategy plugin entry point

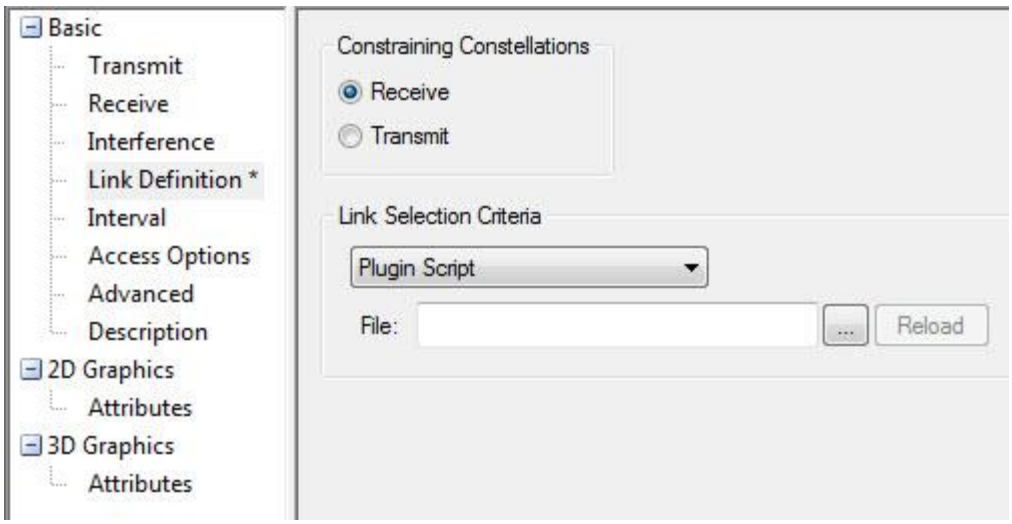


Figure 13 Satellite Selection Strategy plugin entry point

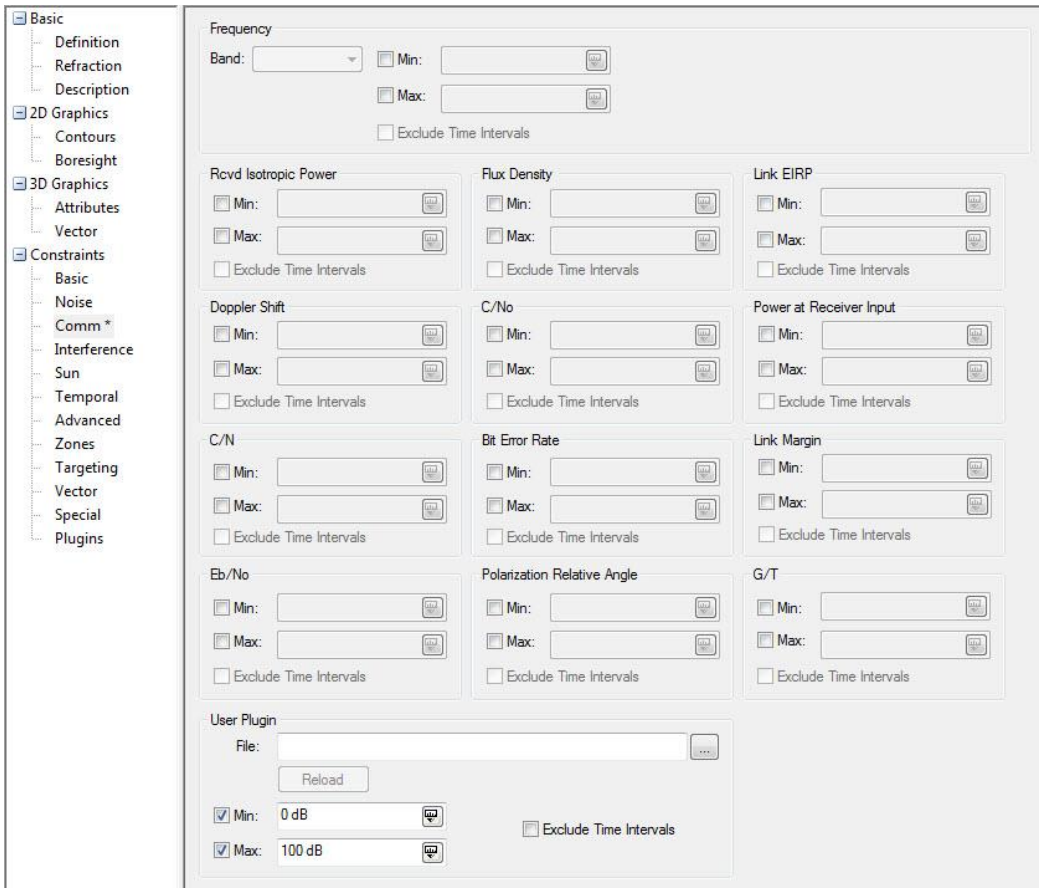


Figure 14 Comm Constraint plugin entry point

Required Plugin File Location: No special directory required. The recommended location is the scenario directory.

Required Module: STK Communications

Appendix B: Plugin Setup for VBScript

- **VBScript**
  - VBScript is automatically installed with Microsoft Windows, but you may require an update to the file ATL.dll. The latest version can be downloaded at http://activex.microsoft.com/controls/vc/atl.cab. Download and save the file, then unzip it. Run the ATL.exe which will automatically put an updated ATL.dll in your Windows System directory.
  - Once the software is installed, register AgScript.dll by opening a command prompt ('DOS window') and changing directories to:
    < **STK install folder** >**\bin\...**
  - Type and run the following command: **regsvr32 AgScript.dll**.
  - You should get a confirmation window that says "DllRegisterServer in AgScript.dll succeeded". Try running your VBScript again.

Appendix C: Tips

- Start with a plugin that you know works. This will verify your licensing and setup.
- Scan the Message Viewer for error messages.
- For optimal performance, turn off plugins when not needed. Be aware that plugins at the application level affect all users and all scenarios.
- Plugins use internal STK units (meters, seconds, radians). The variable that displays in the access constraint panel has units defined by the script.
- Connect commands cannot run from within a plugin. Some plugin properties that you can set in the GUI, such as access constraint thresholds, can be set with Connect.
- Vector Geometry Tool arguments are available for access constraint plugins.
- The name of the plugin function and that of the file itself must match.
- The return variable must be in the form of an array.

Appendix D: Plugin Samples

Samples available at:

- <STK install folder> \CodeSamples\Extend\PluginScripts\….
- <STK install folder>\Data\Resources\stktraining\text\scripting\….
- Advanced Training classes “Integrating & Customizing STK”