

Solar System's Edge

AN ADVANCED KSP TRAJECTORY OPTIMIZATION TOOL TUTORIAL
ARROWSTAR

Contents

Revision History	3
Introduction	4
Tutorial Goals	4
Prerequisites	4
Basis of Mission Design	4
Tutorial Part 1 – Mission Setup	5
Setting Up KSPTOT	5
Using Multi-Flyby Maneuver Sequencer to Develop the Mission Plan	5
Setting Up Multi-Flyby Maneuver Sequencer	5
Running Multi-Flyby Maneuver Sequencer	7
Tutorial Part 2 – Mission Architect Setup	10
Setting up Mission Architect	10
Setting up the Rendezvous Spacecraft	10
Setting up the Comm Relay Spacecraft	10
Verifying the Location of the Kerbal Space Center	11
Final Setup	11
Tutorial Part 3 – Departing Kerbin	12
Setting Up the Kerbin Parking Orbit	12
Create the Departure Sequence	13
Create the Coast to Burn	13
Create the Departure Burn	14
Create Coasts to Eve Periapsis	16
Optimize the Departure Sequence	18
Optimizing the TEI Burn Plan – Round 1	18
Optimizing the TEI Burn Plan – Round 2	20
Optimizing the TEI Burn Plan – Round 3	23
Lessons Learned	25
Tutorial Part 4 – Eve Powered Flyby, Eve Departure, and Jool Arrival	27
Creating the Powered Flyby	27
Optimizing the Eve Departure Burn	29
Optimizing the Eve Flyby Burn – Round 1	30

Optimizing the Eve Flyby Burn – Round 2	32
Lessons Learned	33
Tutorial Part 5 – Powered Jool Flyby, Jool Departure and Plock Arrival.....	35
Creating the Powered Flyby	35
Optimizing the Jool Flyby Burn – Round 1	38
Optimizing the Jool Flyby Burn – Round 2	40
Lessons Learned	42
Tutorial Part 6 – Plock Arrival and Rendezvous Activities	43
Setting Up the Rendezvous	43
Setting Up the Rendezvous Maneuver Planner	43
Targeting the Transfer Orbit	46
Targeting the Rendezvous	47
Lessons Learned	50
Tutorial Part 7 – Analysis	51
Rendezvous Verification	51
Communications Network Analysis	52
Appendix A – Glossary of Terms	55
Appendix B – MFMS Results	56

Revision History

[illegible]

Introduction

Tutorial Goals

The KSP Trajectory Optimization Tool (KSPTOT) is a powerful piece of mission design software intended to make planning space missions in Kerbal Space Program (KSP) as straight forward as possible. In particular, the Mission Architect code allows KSP mission designers to have full freedom and flexibility when putting together mission plans. However, to date the only tutorials that exist for KSPTOT Mission Architect involve single transfers from Kerbin to Duna without taking advantage of more advanced piloting techniques such as flybys of intermediate planets. This tutorial is designed to rectify that problem.

The goal of this tutorial is to develop a mission plan in KSPTOT Mission Architect that departs from Kerbin, flybys of Eve and Jool as gravity assist maneuvers, and arrives at “Plock,” a Pluto-analog body added by Outer Planets Mod.

This tutorial is completed in Mission Architect. The skills that will be exercised in this tutorial include:

- Planning powered and unpowered flybys of planets in gravity assist maneuvers;
- Planning rendezvous with other spacecraft;
- Determining ideal pre-departure parking orbits; and
- Analyzing communication links.

This tutorial assumes basic working knowledge of the KSPTOT Multi-Flyby Maneuver Sequencer tool and Rendezvous Maneuver Planner tool, as well as working knowledge of Mission Architect.

Prerequisites

To complete this tutorial, you will need the following:

1. Kerbal Space Program 1.1.2 or later;
2. [Outer Planet Mod](#) (OPM) v2.0 or later;
3. A KSPTOT bodies.ini file for OPM, see below;
4. KSPTOT v1.5.4 or later; and
5. MATLAB Compiler Runtime (MCR) for your version of KSPTOT (R2015b for KSPTOT v1.5.4).

An example KSPTOT bodies.ini file for this tutorial (item 3 above) is included with this tutorial package.

Basis of Mission Design

Before starting a mission like this, it is required that the Multi-Flyby Maneuver Sequencer (MFMS) is used to find a combination of flyby bodies that yield low required delta-v and the departure and arrival dates for each body in the mission plan. The first part of this tutorial explains how to use MFMS to generate the basic mission plan which will then be implemented in detail in Mission Architect.

Tutorial Part 1 – Mission Setup

Setting Up KSPTOT

To begin, we need to initialize KSPTOT with the correct bodies.ini file. Do the following:

1. Start KSPTOT by double-clicking on the application. Wait for the application to load and present you with the pork chop plot “Main UI”.
2. Load the bodies_OPM.ini file included with this tutorial:
 - a. File menu -> Load Bodies From File;
 - b. Select the bodies_OPM.ini file in the dialog box that appears;
 - c. Select “Open” and wait for KSPTOT to ingest the file.

Note that the time system in use in this tutorial is **Earth Time!** All dates are provided in this system. For the sake of consistency, you should switch over to Earth time now using the menu “Edit -> Time System.”

Using Multi-Flyby Maneuver Sequencer to Develop the Mission Plan

Setting Up Multi-Flyby Maneuver Sequencer

Open up the Multi-Flyby Maneuver Sequencer from the main KSPTOT window. Use the menu “Tools -> Maneuver Planning -> Multi-Flyby Maneuver Sequencer.” You will be presented with the main MFMS user interface.

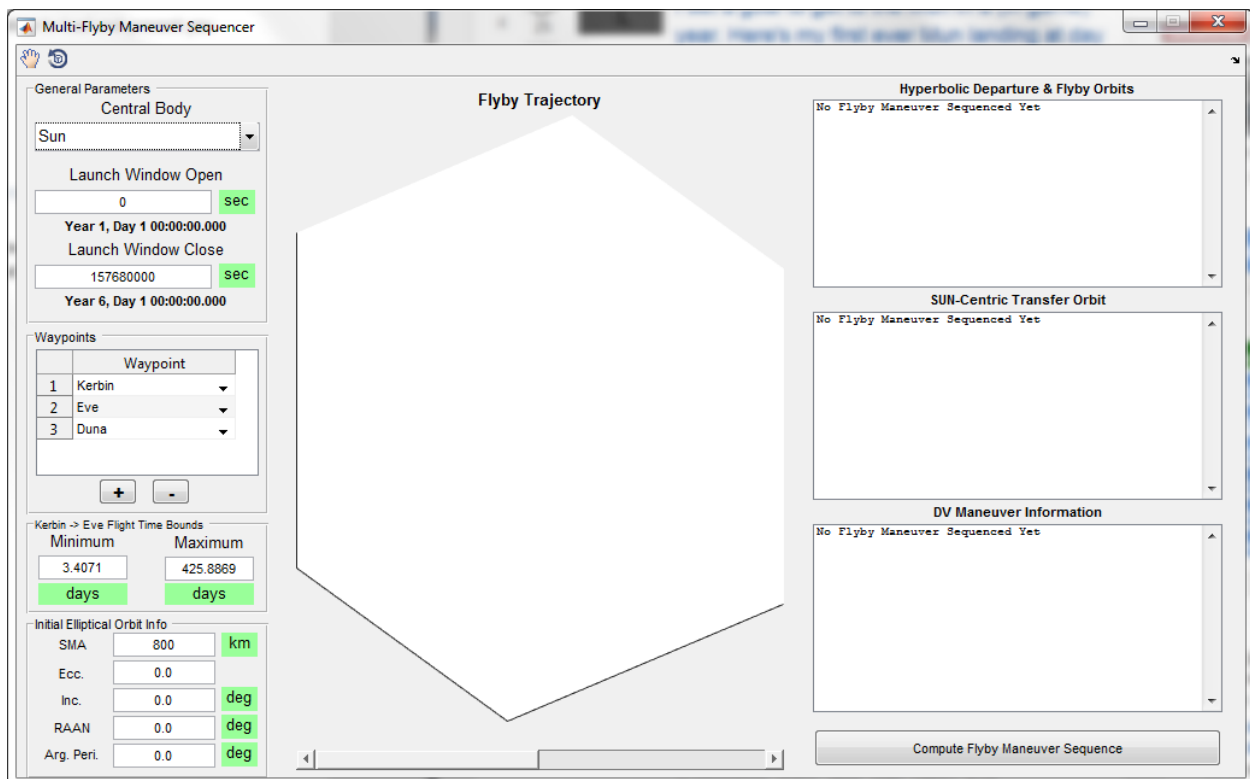


Figure 1: Blank Multi-Flyby Maneuver Sequencer Interface

Recall from the basic description of our mission plan that we wish to depart from Kerbin, use Eve and Jool as gravity assist bodies, and arrive at Plock. To begin, ensure that the central body input at the top left part of the screen is set to “Sun.”

The “launch window open/close” parameters define the range of acceptable points in time that the spacecraft may leave the first waypoint in your waypoint list. The optimizer will select the actual launch date in such a way that the total mission delta-v cost is minimized. For now, leave these parameters at their default values, which should be 0.0 seconds for the launch window open (Year 1, Day 1) and 157680000.0 seconds for the launch window close (Year 6, Day 1). This corresponds to a 5-year period during which we will look for a Kerbin departure date.

Next, move to the Waypoints section of the user interface. The first waypoint in the list is always your departure body, the place you are starting from. The last waypoint in the list is always your arrival body, the place you wish to end up. Any intermediate bodies will be used for gravity assist maneuvers.

We have four waypoints in this mission plan but only three rows in our waypoints table right now. Push the “+” button at the bottom of the waypoints list to add a fourth waypoint. Afterwards, set the waypoints list such that waypoint 1 is Kerbin, waypoint 2 is Eve, waypoint 3 is Jool, and waypoint 4 is Plock. When you are finished, your waypoints list should look like Figure 2.

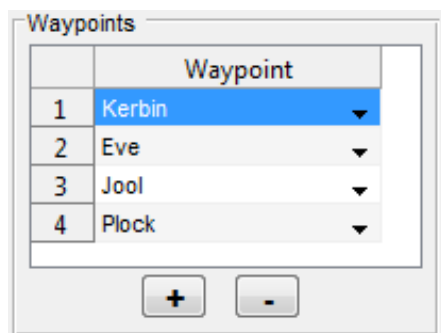


Figure 2: Completed MFMS Waypoints List

Below the waypoints list is an area for “Flight Time Bounds.” This allows you to specify the minimum and maximum amount of time the optimizer will allow when looking for trajectories between bodies in the waypoints list. As an example, we will alter two sets of bounds here.

Select the first row in the waypoints list, the one that shows “Kerbin.” Notice when you select this row, the Flight Time Bounds values change and the title of the box changes to “Kerbin -> Eve Flight Time Bounds.” If you’d like, select another row in the waypoints list to watch how the values in the Flight Time Bounds area change. When you are finished, select Kerbin again in the waypoints list.

In the Flight Time Bounds area, ensure that the title of the box shows “Kerbin -> Eve Flight Time Bounds.” Then, update the Minimum box with the value 20 days and the Maximum box with a value of 400 days. Push enter to commit the changes.

Next, click on Jool in the waypoints list in order to bring up the “Jool -> Plock Flight Time Bounds.” Set the lower bound to 500 days and the upper bound to 7000 days. It’s a long way from Jool to Plock!

Notice that if you select Plock in the waypoints list, the Flight Time Bounds gray out. This is because the first planet in each transfer arc is used to set the flight time bounds.

Finally, we can adjust the initial orbital information in the box in the lower left. Change the SMA value here to 700 km and leave the rest of the parameters as they are (the rest should all be zero).

And that’s it, we’ve successfully set up MFMS for our run. When you are finished, the user interface should look like Figure 3.

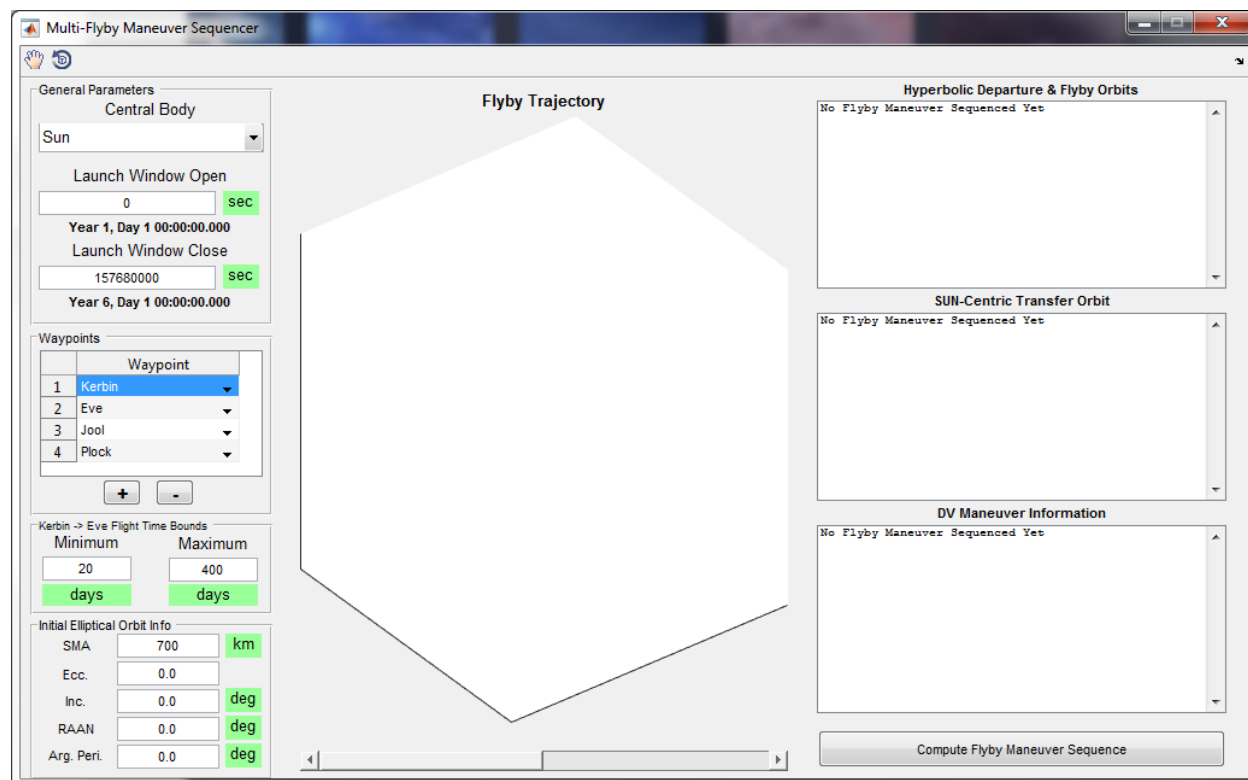


Figure 3: Multi-Flyby Maneuver Sequencer UI Setup

Running Multi-Flyby Maneuver Sequencer

The MFMS is based upon an optimization technique known as a “genetic algorithm.” Fundamentally, this works by defining a large number of possible trajectories that fit our mission plan, evaluating how much delta-v each costs to accomplish, and then “breeding” the various trajectories together in such a way that, in theory, generates lower cost mission plans with less delta-v required to accomplish. Both the “breeding” and the initial set of trajectories are selected randomly, introducing a non-deterministic quality to MFMS, which is helpful for fully exploring the trade space over multiple runs of the tool. The actual details of how the genetic algorithm works, of course, are significantly more complicated but that is the idea summarized.

Start MFMS running by pushing the “Compute Flyby Maneuver Sequence” in the lower right corner. The code will initialize the optimizer and begin searching for an optimal trajectory. You should see a window that presents you with the status of the run, similar to that shown in Figure 4.

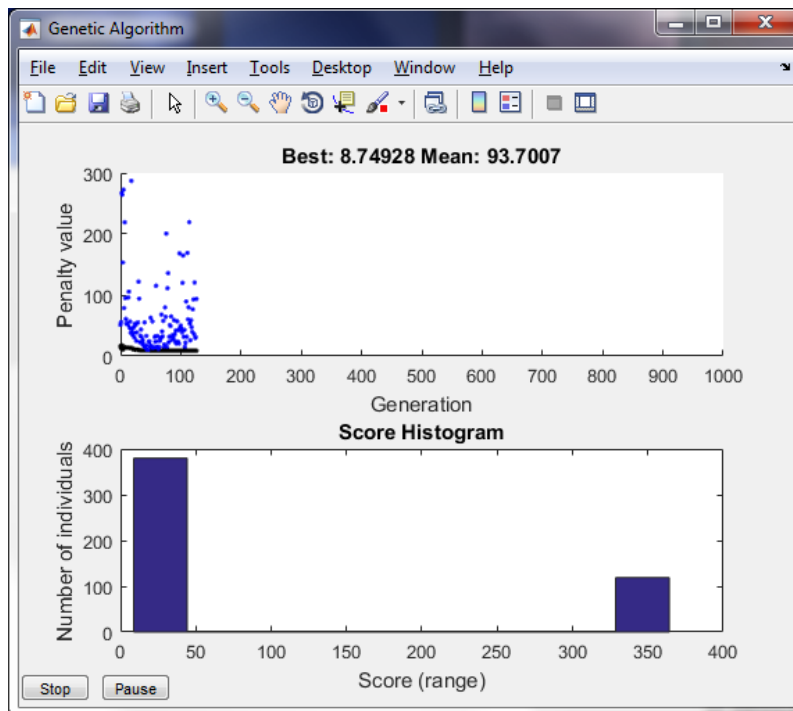


Figure 4: MFMS Genetic Algorithm Status Window

When the code finishes, the central area will show a map of the flight plan through the various waypoints. The right-hand side will provide information on the planet-based inbound and outbound orbits, the various transfer orbit between planets, and the delta-v maneuvers required at each planet to accomplish the mission. You should get something like that shown below in Figure 5.

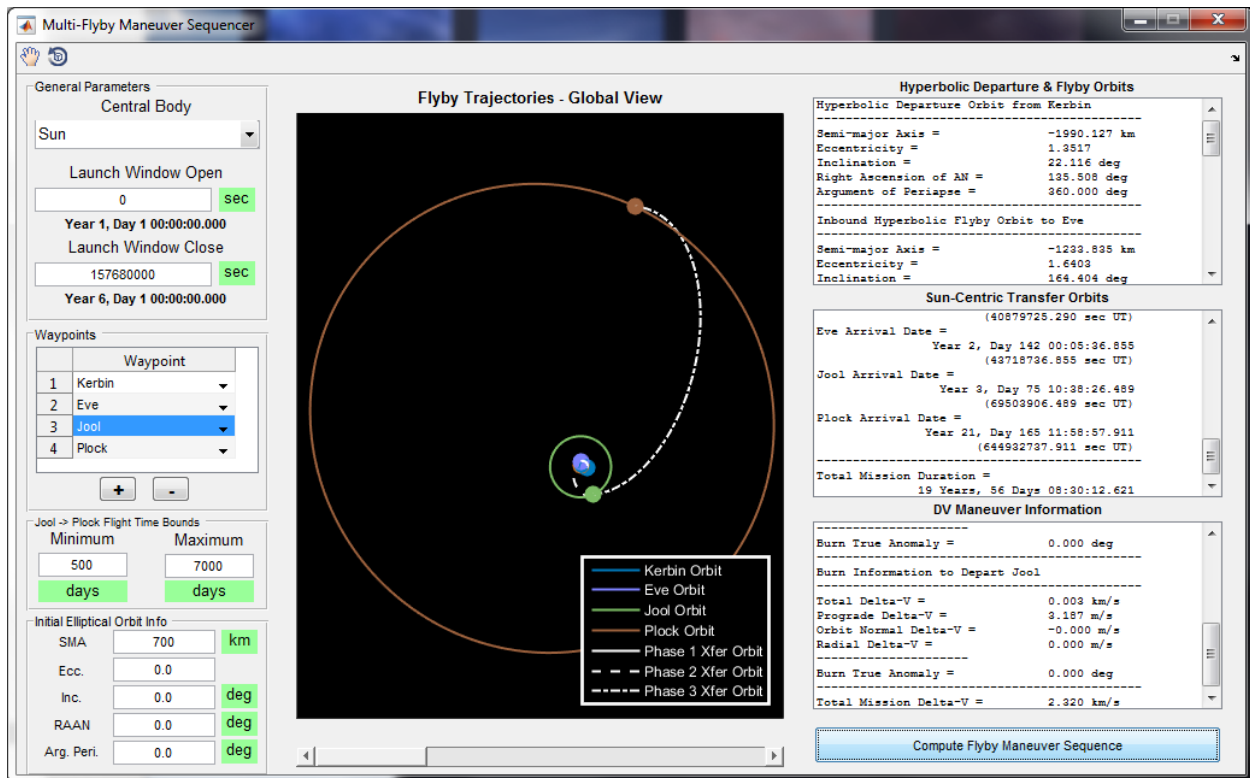


Figure 5: Completed Multi-Flyby Maneuver Sequencer Run

If you'd like, go ahead and push "Compute Flyby Maneuver Sequencer" a few more times, recording the details of each run as you do so (the text in the right-hand side boxes). After a few runs, evaluate the total mission delta-v as shown at the bottom of the "DV Maneuver Information" box and pick out the mission with the smallest delta-v cost.

As mentioned before, MFMS works off of a highly efficient genetic algorithm for trajectory optimization. Therefore, the results it produces are non-deterministic and tend to be different each time you run the program, even if only slightly. To compensate for that in this tutorial, this mission plan will start with results from MFMS I previously generated. These results are provided in "Appendix B – MFMS Results" at the end of this document. Please take a look at these results now and perhaps copy them somewhere else so they will be readily available to you when you need them in the following Mission Architect sections.

Tutorial Part 2 – Mission Architect Setup

Setting up Mission Architect

In this section, we're going to set up our target "rendezvous" spacecraft, set up our comm relay spacecraft, and verify that the Kerbal Space Center (KSC) is located on Kerbin at the right location.

To begin, start KSPTOT Mission Architect from the Tools menu.

Setting up the Rendezvous Spacecraft

Do the following:

1. Open the Other Spacecraft menu: Spacecraft -> Other Spacecraft...
2. Push the "Add Spacecraft" button to create a new Other Spacecraft.
3. Rename the spacecraft from "New Spacecraft" to "Plock Orbiter 1."
4. Change the line color from Red to Orange.
5. Change the line style from Solid Line to Dashed-Dot Line.
6. Change the orbit's central body from Sun to Plock.
7. Set the orbit up as follows:

Quantity	Value	Unit
Epoch	0.0	sec
Semi-major Axis	400.0	km
Eccentricity	0.0	
Inclination	0.0	deg
RAAN	0.0	deg
Argument of Periapsis	0.0	deg
True Anomaly	0.0	deg

8. Set the Comm Max. Range to 612846.0564 km. This is the radius of the Plock sphere of influence (SOI).
9. Leave the Other Spacecraft dialog open for the next steps.

Setting up the Comm Relay Spacecraft

Do the following:

1. Push the "Add Spacecraft" button to create a new Other Spacecraft.
2. Rename the spacecraft from "New Spacecraft" to "Jool Relay 1."
3. Change the line color from Red to Green.
4. Change the line style from Solid Line to Dashed-Dot Line.
5. Change the orbit's central body from Sun to Jool.
6. Set the orbit up as follows:

Quantity	Value	Unit
Epoch	0.0	sec

Semi-major Axis	35750.0	km
Eccentricity	0.81818182	
Inclination	90.0	deg
RAAN	0.0	deg
Argument of Periapsis	90.0	deg
True Anomaly	0.0	deg

7. Leave the Comm Max. Range at “Inf”, for infinite range.
8. Close the Other Spacecraft dialog.

Verifying the Location of the Kerbal Space Center

Do the following:

1. Open the Ground Targets menu: Spacecraft -> Ground Targets
2. Verify that there exists an entry in the list called “KSC” with the following parameters:

Quantity	Value	Unit
Location Body	Kerbin	sec
Longitude	285.4247	degE
Latitude	-0.1025	degN
Altitude	0.06841	km

3. Set the Comm. Max Range to 65334883 km, which is the approximate radius of apoapsis of Jool’s orbit.
4. Close the Ground Targets dialog box.

Final Setup

It is good practice to save the mission plan you are working on often. At this point, let’s go ahead and save the plan. Use the File -> Save Mission Plan As... menu to save. Select a convenient folder to put the mission plan “*.MAT” file. Call the file name “SolarSystemsEdge.mat”.

That’s it! We’re all set up and ready to begin the actual mission planning now. You may now move on to Part Three of this tutorial.

Tutorial Part 3 – Departing Kerbin

During this phase of the mission design process, we will design our Kerbin parking orbit, create the initial mission plan to take us to Eve, and then optimize that plan to achieve the desired orbital parameters at Eve.

Setting Up the Kerbin Parking Orbit

The principle requirement of a parking orbit to be used as a starting point for an interplanetary journey is that the plane of the parking orbit should match the plane of the hyperbolic departure orbit. This eliminates out of plane components of delta-v and typically also eliminates radial components, leaving the delta-v vector strictly along the velocity vector, which is more efficient. In practice this means the two orbits have matching values of inclination, RAAN, and argument of periapsis.

Consider the hyperbolic orbit we were given from MFMS, as shown above and copied below.

Table 1: Hyperbolic Departure Orbit from Kerbin

Hyperbolic Departure Orbit from Kerbin	

Semi-major Axis =	-2004.600 km
Eccentricity =	1.3991
Inclination =	21.119 deg
Right Ascension of AN =	137.848 deg
Argument of Periapse =	360.000 deg

In order to satisfy the “principle requirement,” let us modify the initial state of our mission plan to use the inclination, RAAN, and argument of periapsis as shown above. Double click on the event called “Initial State” in the mission script list. Modify the three angle parameters (inclination, RAAN, and argument of periapsis) to match what is shown above. Additionally, set the SMA of the orbit to 700 km, as this was the initial orbit used in the MFMS run. Leave the eccentricity and true anomaly values as they are (both 0.0).

The mass of the spacecraft matters little here, but we can make life easier on ourselves by selecting optimistic values to do the initial mission design. As your spacecraft design evolves, you may begin substituting those mass values for what we will enter here. For now, use a dry mass of 5.0 tons, a Fuel/Ox mass of 15.0 tons, and zero tons of monopropellant and xenon.

Finally, we need to update the orbit epoch. Recall that the Kerbin departure date is as is shown previously and copied below.

Table 2: Kerbin Departure Date

Kerbin Departure Date =
Year 2, Day 109 10:25:19.242
(40904719.242 sec UT)

Go ahead and set the epoch of the initial state to that of the departure epoch. Even though we may be off by a few minutes or hours, in the grand scheme this is very small.

Once you've completed this, save and close the initial state. The dialog box should appear identical to Figure 6.

State Name	
Initial State	

Orbit	
Orbit	Kerbin
Epoch	40904719.242 sec
SMA	700.0 km
Ecc.	0
Inc.	21.119 deg
RAAN	137.848 deg
Arg. Peri.	0.0 deg
True Anom.	0 deg

Mass	
Dry Mass	5.0 tons
Fuel/Ox	15.0 tons
Monoprop	0.0 tons
Xenon	0.0 tons

Save & Close Cancel

Figure 6: Mission Plan Initial State

Create the Departure Sequence

A departure sequence always consists of two events: a coast to the burn true anomaly and delta-v maneuver. Let's create these now.

Create the Coast to Burn

First, insert a Coast by using the "Select Event to Insert" menu and selecting "Coast." You will be presented with the "Create Coast" dialog box.

Start by renaming the Coast to "Coast to Trans-Eve Inject". Leave the burn type as "Go to True Anomaly" and leave the color as Red.

The maneuver planning information we have for this burn does actually include information regarding the burn true anomaly. However, this information is based on the default MFMS equatorial Kerbin parking orbit, which we are not using here. In actuality, we don't have any usable information on where the burn will be located. Therefore, we should guess a value that is in the middle of our upper and lower bounds to allow the optimizer to search well in both directions. In this case, set the bounds to -180 degrees and 360.0 degrees and set our burn true anomaly to 0.0 degrees to begin. Make sure the "Opt?" checkbox is selected as the optimizer will need to adjust this value.

Finally, we're going to leave the "Revs Prior to Coast" value at 0 as there is no need to go around the body more than once. The reference body may be left blank as well.

When you've finished, you may save and close the dialog. The coast parameters should look like Figure 7.

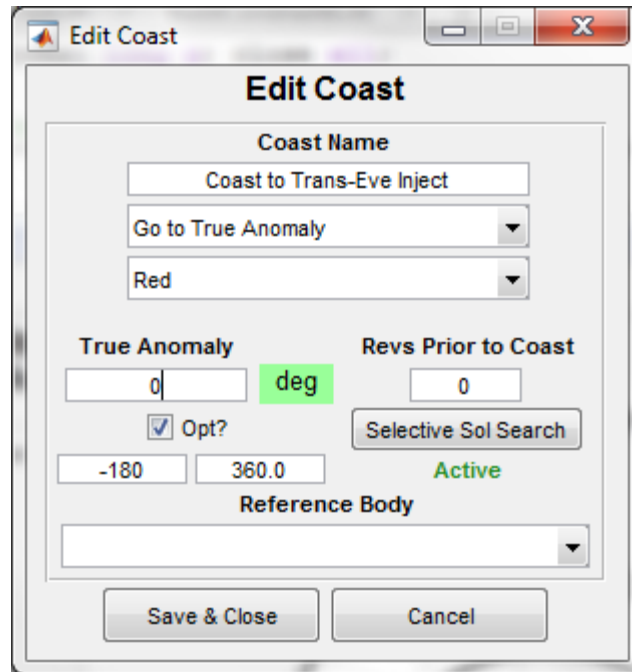


Figure 7: Coast to Trans-Eve Inject

Create the Departure Burn

Insert a “Delta-V Maneuver” into the mission script in a manner similar to how you created the coast above. You will be presented with the “Create DV Maneuver” dialog box.

Rename the maneuver “Trans-Eve Injection Burn” and leave the burn type as “Proscribed Delta-V (Orbital Vector)”. This means that the burn delta-v is proscribed (applied instantaneously) and the direction of the burn is specified in “prograde”, “normal”, and “radial” components. The engine selection is immaterial at the moment, so leave it at the default selection.

Actually setting up the delta-v vector can be a bit tricky. The information we have from MFMS for this burn is shown below.

Table 3: Burn Information to Depart Kerbin

Burn Information to Depart Kerbin	

Total Delta-V =	1.500 km/s
Prograde Delta-V =	934.697 m/s
Orbit Normal Delta-V =	1172.544 m/s
Radial Delta-V =	0.000 m/s

Keep in mind that the Orbit Normal Delta-V shown no longer applies since our parking orbit is already in the plane of the hyperbolic departure orbit. The Prograde Delta-v is also probably going to be larger than what is shown above, but we can use it as an initial guess.

Populate the vector itself with a prograde value of 934.697 m/s and zero for the other two components. Additionally, turn off the Radial optimization flag by unchecking the “Opt?” checkbox. We turn this off because we want to prevent the optimizer from adding radial delta-v that is costly and not very fuel efficient.

Setting the departure burn bounds can also be tricky. For the prograde component, I like to use a lower bound that is greater than zero (meaning the burn has to be prograde, making our SMA grow larger and therefore sending us up and out) and an upper bound which is fairly generous. Here, I’ve selected 500 m/s and 2500 m/s for these values, respectively. Guessing here usually comes down to experience and it is perfectly okay to adjust them if your optimization runs consistently run up against one of these bounds.

For the normal component, we do not expect much, but there may be some small amount required. Using ± 100 m/s for the bounds initially is likely okay to begin.

When you’ve completed entering the values into the dialog box, you can save and close it. It should appear as in Figure 8.

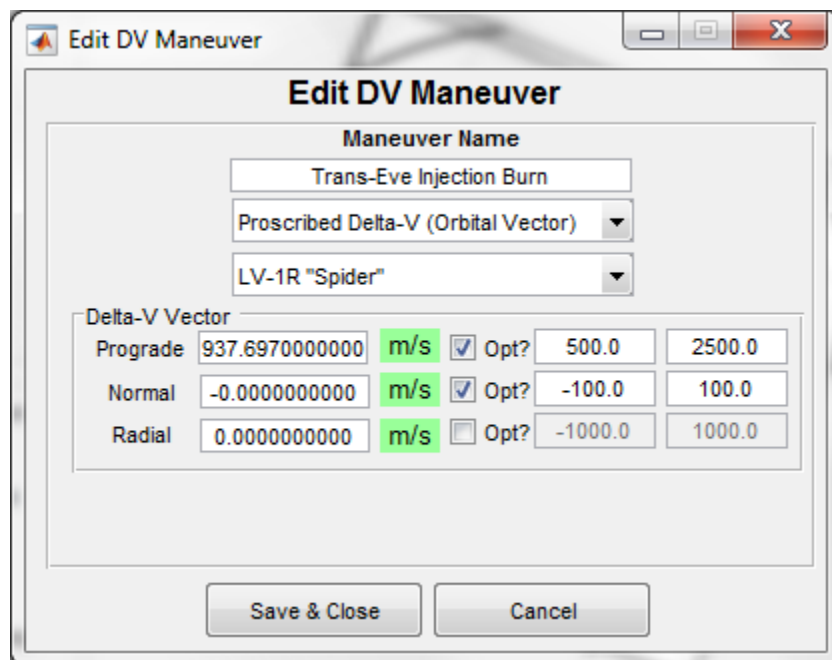


Figure 8: Trans-Eve Injection Burn

Note that post burn state shown in the main Mission Architect window shows the eccentricity of the final orbit as greater than 1.0. This is critical. The optimizer will have a very hard time finding a solution to target Eve if the orbit never leaves the Kerbin system.

Create Coasts to Eve Periapsis

Finally, we need to add coasts to leave the Kerbin system and head to Eve. I like to do this in two stages. First, I add a coast to the next SOI transition, which should be the transition from Kerbin's SOI to the sun's SOI. The second coast takes us from the edge of the Sun's SOI to the periapsis of body that we want to head to. The reason I do this is to allow the second coast, the Sun-centered coast, to be able to apply one "revolution prior to coast." This lengthens the total arc of the coast to its maximum and therefore provides the optimizer with more arc length to work with when attempting to target a burn to hit another body.

To do so, add a coast in a manner similar to the previous coast. Rename the coast to "Coast to Sun SOI" and set the coast type to "Go to Next SOI Transition." Change the coast color to Orange. Save and close the coast when you have finished. It should look like Figure 9.

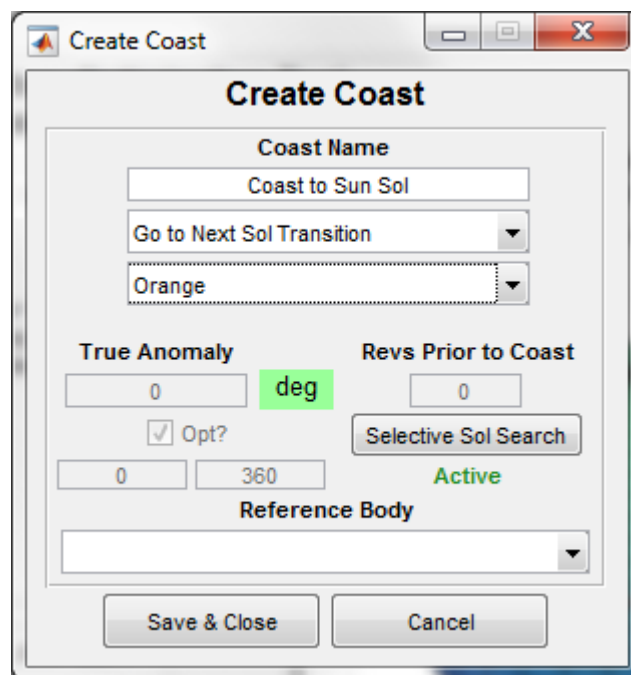


Figure 9: Coast to Sun SOI

Note that your Final Spacecraft State orbit is now around the Sun and not Kerbin.

Next, add another coast. Rename it to "Coast to Eve Peri" and set the coast type to "Go to Periapsis." Set the coast color to Yellow. **Importantly**, set the Reference Body of this coast to Eve.

We also need to set the Selected SOI search, which should be shown in green text as active. Push the Selected SOI Search button. All the bodies will be selected. Simply modify the selection so that only the Sun, Eve, and Gilly are selected. This speeds up the SOI search algorithm by having it ignore bodies that we don't wish to consider. In this case, the coast will only search for SOI transitions to the Sun, Eve, and Gilly. Gilly is included to make sure we capture any (unlikely) Gilly transits on our inbound trajectory. To

select multiple bodies, hold down the control key and click on the three body names in the list. When you have the three bodies selected, push Okay.

Finally, I like to set the Revs Prior to Coast for coasts which will be searching for a body to intersect to 1. This maximizes the area of the orbit that the optimizer can use to find the minimum distance to Eve. We will eventually set this value to 0 after we find our Eve intersect, but for now, set it to 1.

When you are finished modifying the coast, save and close the coast. It should look like Figure 10.

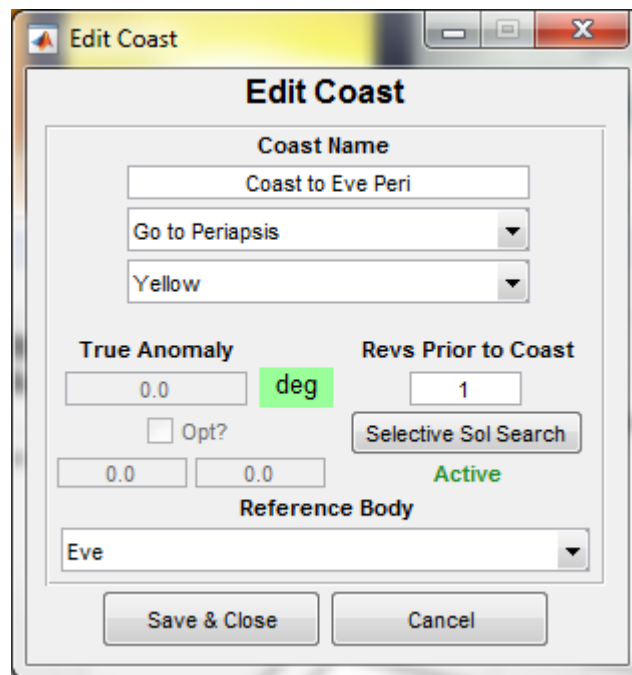


Figure 10: Coast to Eve Peri

Your final Mission Architect window should look something like Figure 11 after all these steps. Importantly, notice that there are asterisks (*) next to Event 2 and Event 3 in the mission script. This means that these events have parameters on them which will be modified whenever the optimizer is run.

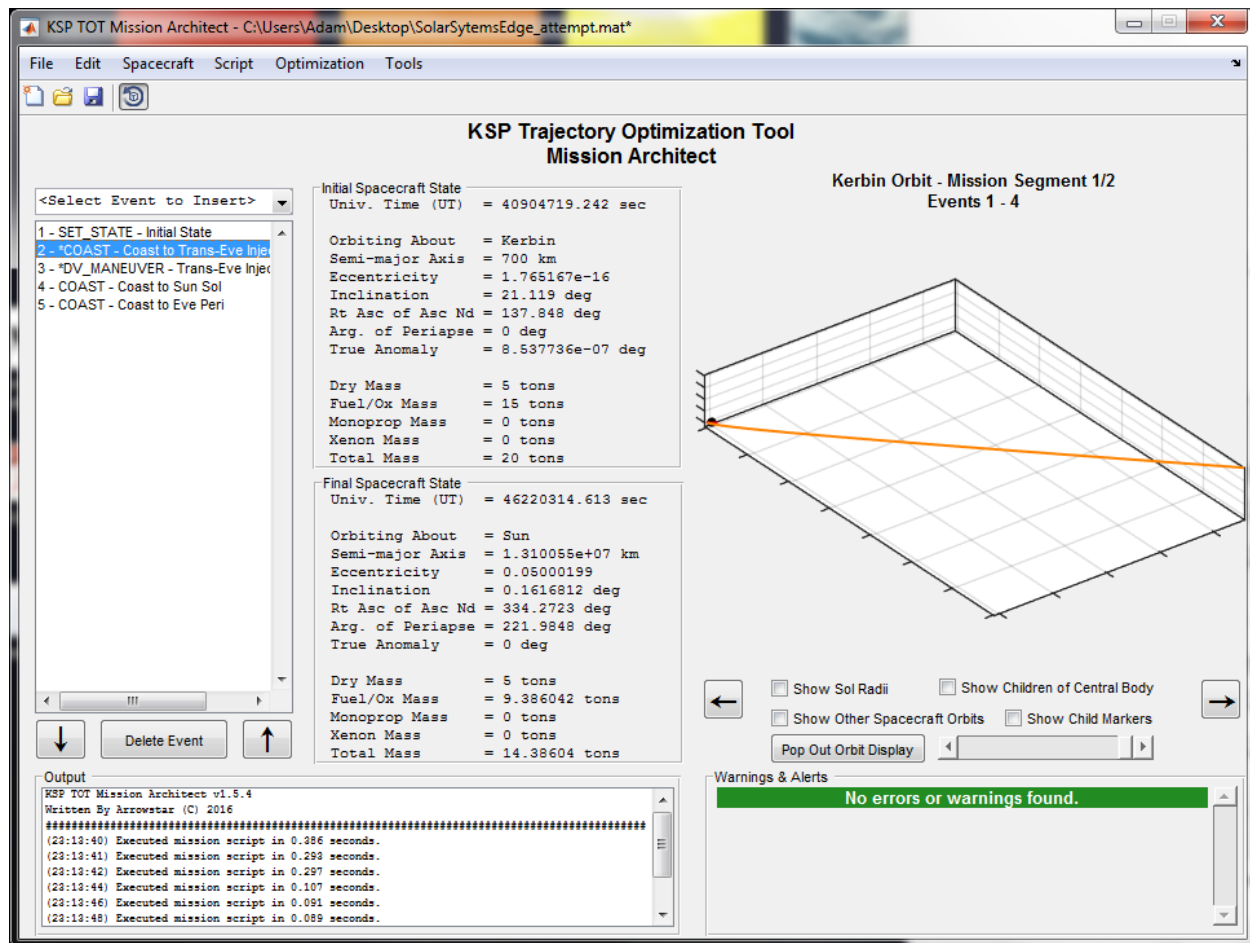


Figure 11: Mission Architect Window - Pre-TEI Burn Optimization

Optimize the Departure Sequence

It's worth a brief pause at this junction to discuss what we need to do next. Right now we have a mission plan that does not take us to Eve. In fact, it probably gets nowhere near Eve. The purpose of using the optimizer, then, is to modify the trajectory in such a way that we are headed on a path towards Eve. Contrary to what it may sound like, we are not making the mission plan "more optimal" by doing so. Instead, we are using the optimizer to goal seek, to help us accomplish something we wish to achieve.

With that out of the way, let's continue. Before we optimize, be sure to save your progress here. Saving prior to an optimization run is useful in the rare event that between you and the optimizer something gets irreversibly fouled up.

Optimizing the TEI Burn Plan – Round 1

Open the Mission Optimizer by using the Optimization menu -> Optimize Mission selection. First, we select the objective function to "minimize distance to body." This will instruct the optimizer to try to adjust our trajectory to minimize the distance to whatever celestial body we select. Select Event 5, Coast to Eve Peri, as the Applicable Event, and select Eve as the Applicable Body.

For now, we're simply going to run with no constraints. The goal is to make it as easy as possible for the optimizer to find Eve. Afterwards, we'll set a constraint on the arrival time and orbit and rerun the optimizer.

Your Mission Optimizer window should look like Figure 12 before you run.

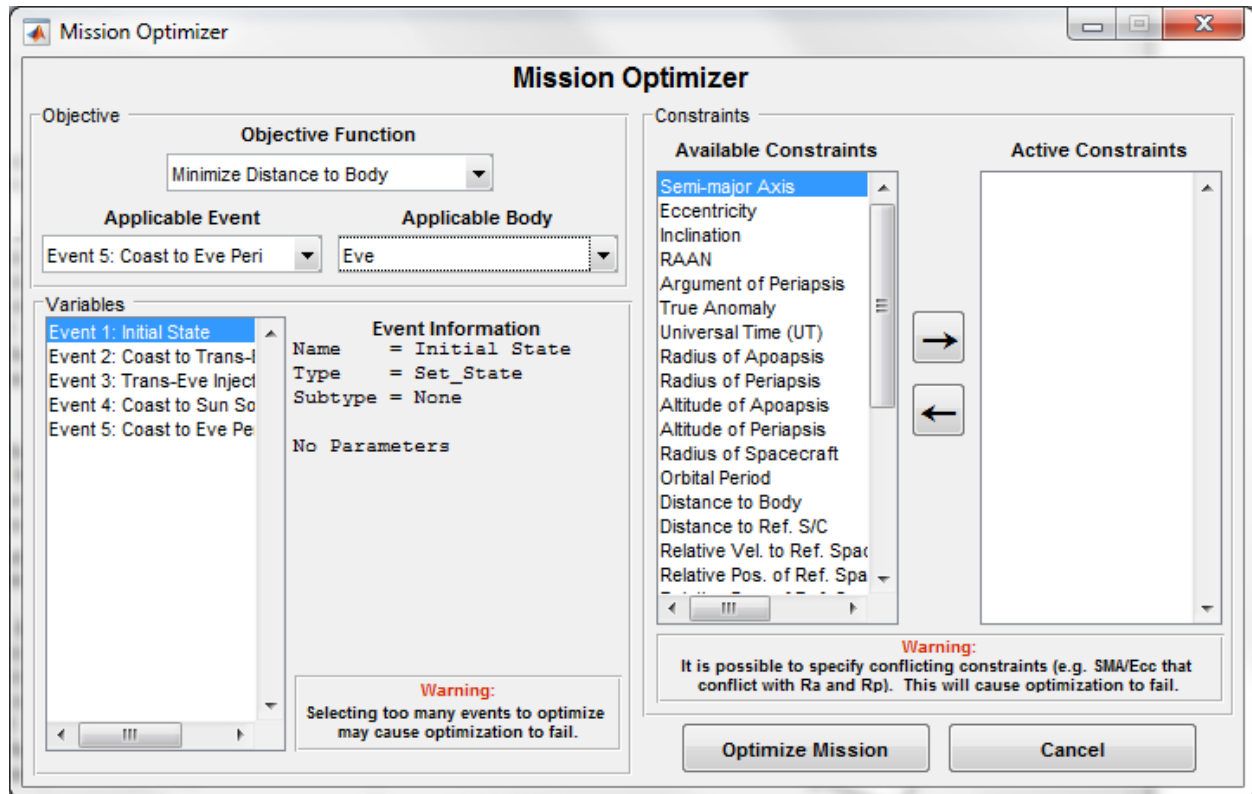


Figure 12: Mission Optimizer - TEI Burn Round 1

When you are ready, push the Optimize Mission button. The optimizer will start up and begin varying the coast and burn parameters and it should terminate with the final trajectory going right through Eve.

After accepting the optimization results, your trajectory should look like Figure 13. Note that you'll need to switch to Mission Segment 2 (use the large arrow buttons), turn on Show Children of Central Body, and zoom in.

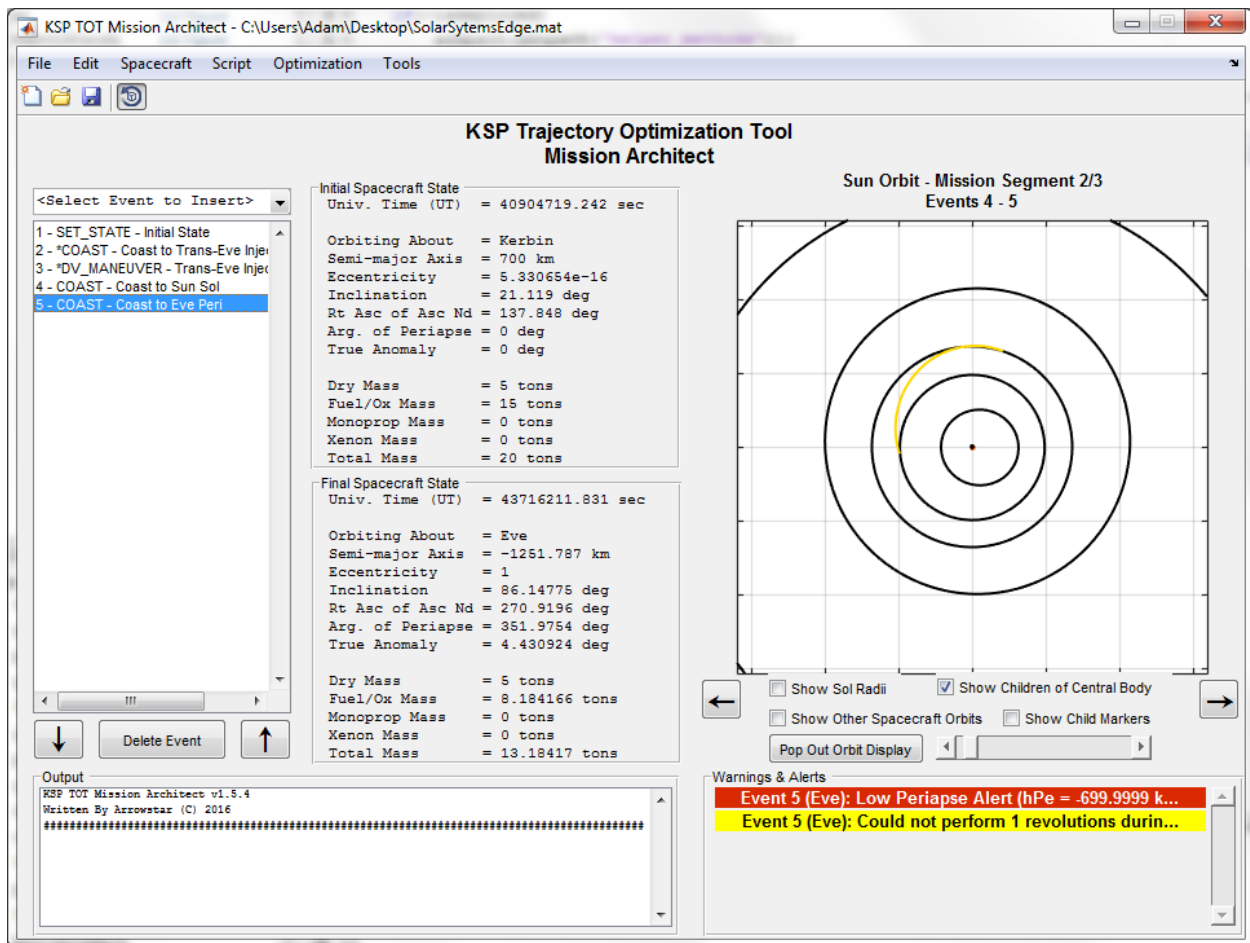


Figure 13: Mission Architect after TEI Optimization Round 1

You'll see we have a problem. Our trajectory goes straight into Eve, as shown by the red Low Periapsis Alert. If we use this trajectory we will crash! Let's use the optimizer to come up with something a bit saner.

Optimizing the TEI Burn Plan – Round 2

Open the Mission Optimizer again. Now that we've made it to the Eve system, let's set some constraints to keep us there and provide a usable orbit. First, I like to set a Central Body constraint on Event 5 to keep us at Eve. I also like to set a Universal Time constraint on Event 5 to make sure we're getting to Eve at the correct time (meaning our trajectory from Kerbin to Eve is close to what MFMS provides). We should try to arrive within 1 hour of our target time, which is shown below.

Table 4: Eve Arrival Date

Eve Arrival Date = Year 2, Day 142 00:27:14.948 (43720034.948 sec UT)

Set the lower bound of the UT arrival constraint to 43716434.948 seconds and upper bound to 43723634.948 seconds. Set the Celestial Body to Eve and the Event to Event 5.

Now we need to constrain the orbit. Our target Eve inbound orbit from MFMS is shown below.

Table 5: Inbound Hyperbolic Flyby Orbit to Eve

Inbound Hyperbolic Flyby Orbit to Eve	

Semi-major Axis =	-1224.660 km
Eccentricity =	1.6451
Inclination =	165.018 deg
Right Ascension of AN =	122.101 deg
Argument of Periapse =	159.994 deg
Periapse Radius =	790.000 km

It's usually easier to do the orbit plane constraints (inclination, RAAN, and argument of periapsis) first and then deal with the SMA and eccentricity constraints in another run. So, add these three angle constraints with a ± 1 degree offset on the upper and lower bounds respectively.

When you've added the constraints, set the objective function to "No Optimization (Satisfy Constraints Only)". This tells the optimizer to ignore any objective function and concentrate on simply making the constraints go away. Your Mission Optimizer window should look like Figure 14.

Note that if you have issues with the optimizer converging (that is, satisfying all the constraints), you can change the objective function back to the "minimize distance to body" function. A good trick for making the optimizer behave without radically changing your optimization setup too much is to alter the objective function.

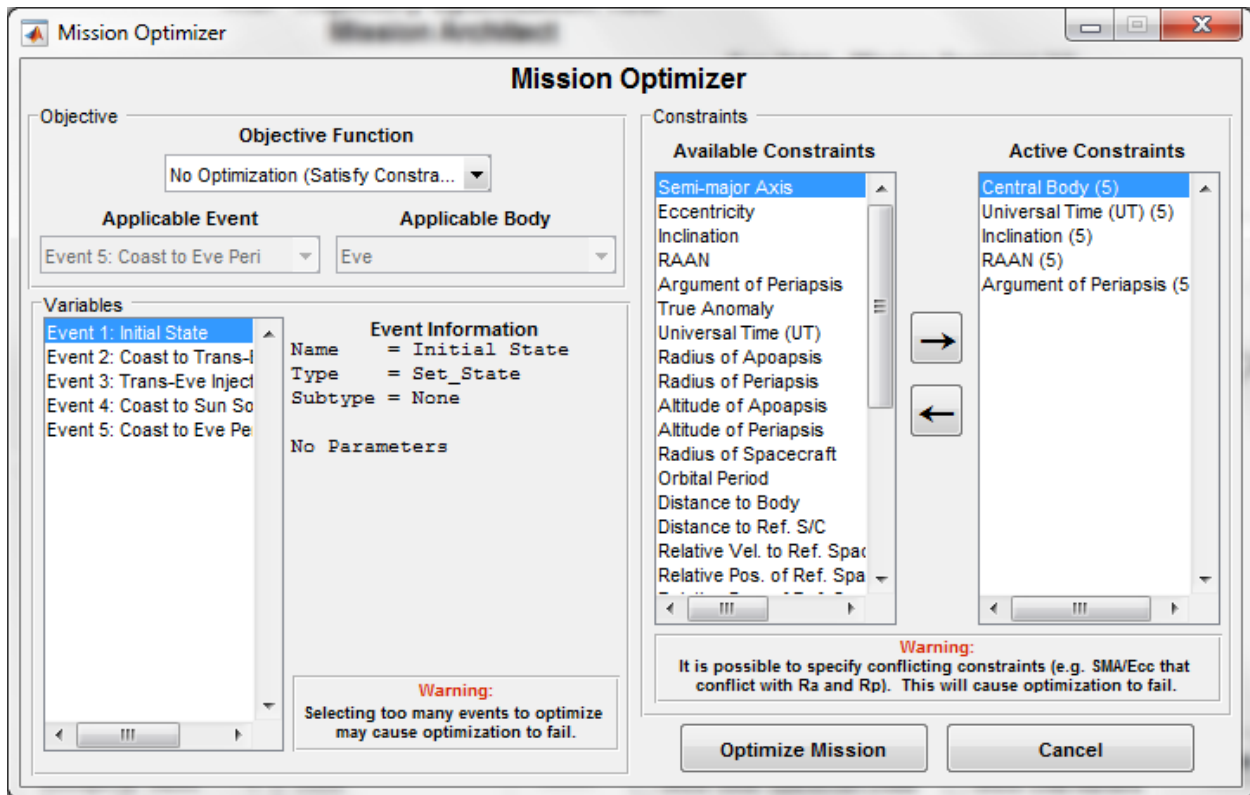


Figure 14: Mission Optimizer - TEI Burn Round 2

When you are ready, push the Optimize Mission button and wait for the optimizer to finish. It should satisfy all the constraints fairly quickly and then finish.

Your Mission Architect window after this round of optimization should look like the following:

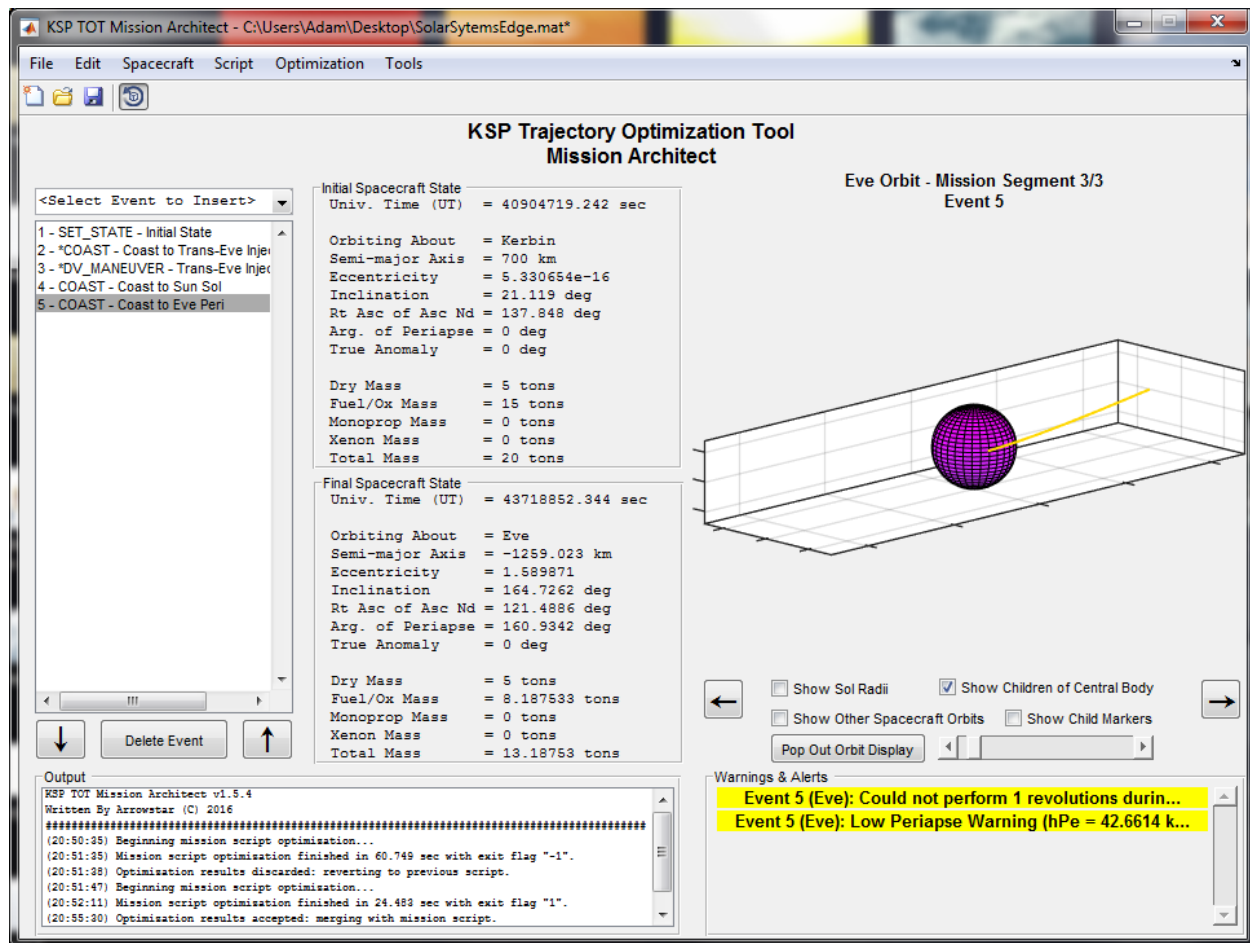


Figure 15: Mission Architect after TEI Optimization Round 2

Finally, we need to use the optimizer to get our semi-major axis and eccentricity correct as well. Don't worry about any warning(s) at the moment, we'll handle those in the next section.

Optimizing the TEI Burn Plan – Round 3

Reopen the Mission Optimizer once more. This time we will add a radius of periapsis constraint and an eccentricity constraint in the same way we added the three other constraints. Use the values in Table 5 above.

Normally when optimizing a mission plan like this, I would go ahead and set a constraint on SMA. However, there are two important considerations here that dissuade me from this course of action. First, the radius of periapsis on this flyby is going to be right above the upper edge of the atmosphere. Setting an SMA constraint may allow the trajectory to go into the atmosphere, whereas a radius of periapsis constraint (which is equivalent in most respects) can be set to forbid this. Second, while putting together this tutorial, I found that the SMA constraint gave the optimizer problems satisfying constraints.

With that foreknowledge, create a radius of periapsis constraint. Set the lower bound to 790 km (the upper edge of the atmosphere) and the upper bound to 810 km, which just provides a 20 km margin for

the optimizer to work in. Note that the atmosphere heights shown here are given as “radius” values, measured from the center of the body. They are not altitudes, which are measured from the surface of the body.

Finally, set an eccentricity constraint with lower bound 1.6441 and upper bound 1.6461.

After adding these two constraints, you Mission Optimizer window should look like Figure 16.

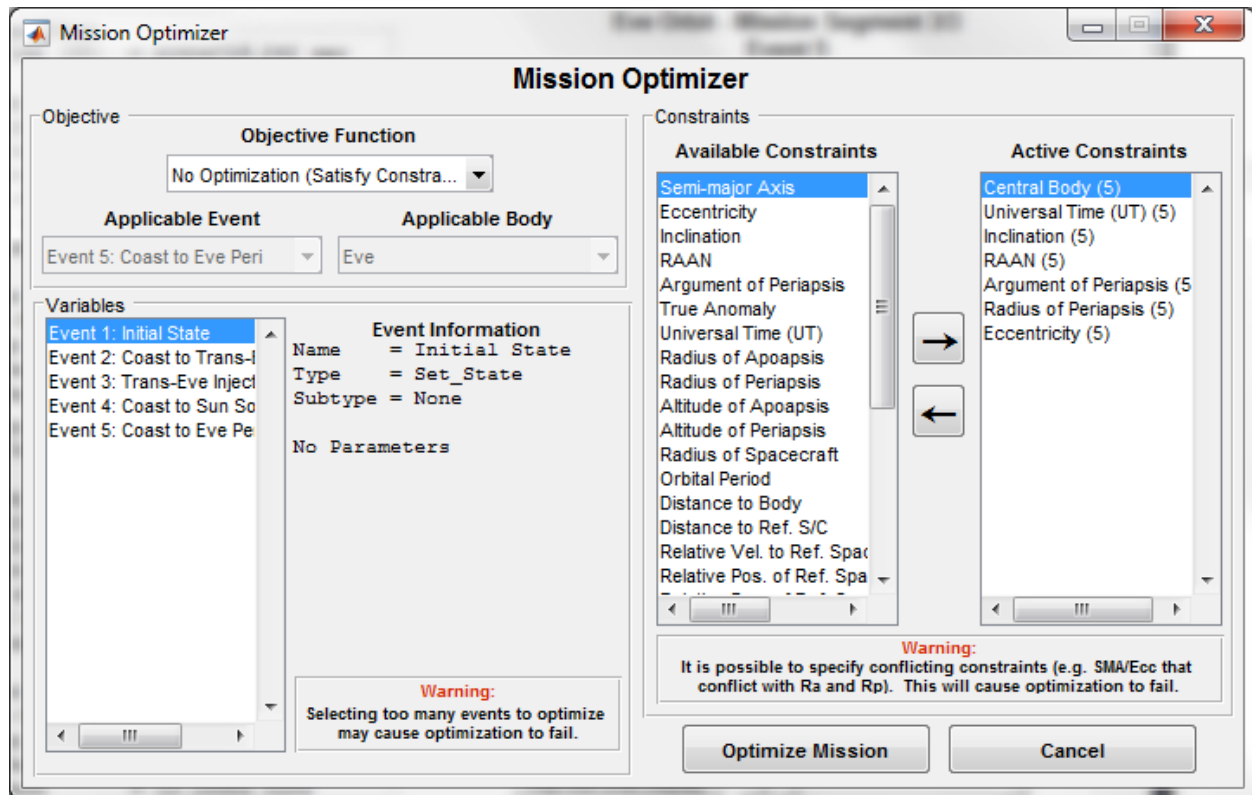


Figure 16: Mission Optimizer - TEI Burn Round 3

Your objective function should still read “No Optimization.” When you are ready, push the Optimize Mission button and wait for the optimizer to converge.

After optimization completes, the main Mission Architect window should look like the following. Notice the final spacecraft state and ensure yours matches. Also notice that the Low Periapsis Alert and Warning have disappeared.

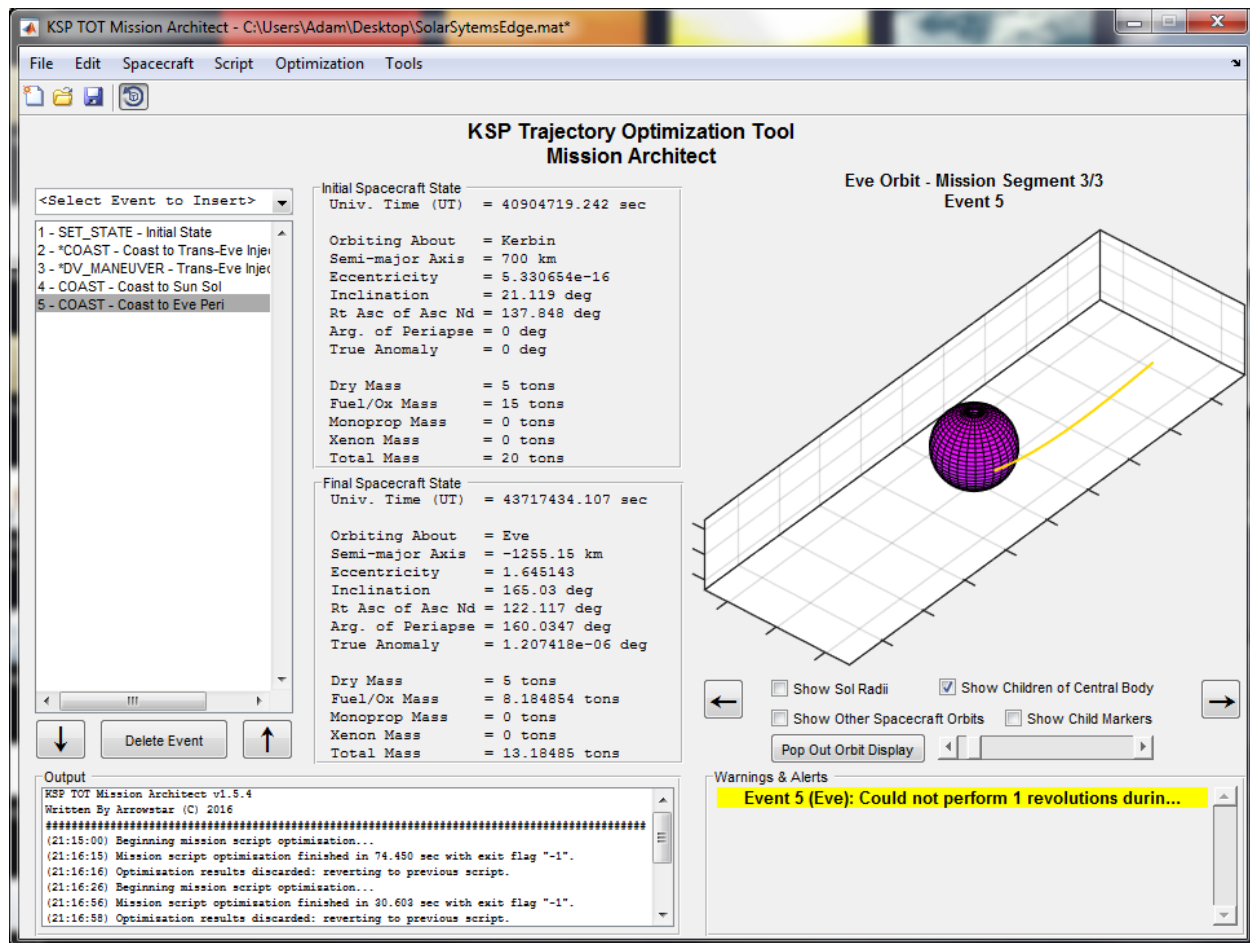


Figure 17: Mission Architect after TEI Optimization Round 3

Finally, let's get rid of that nagging revolutions warning. Double click Event 5 in the mission script, set the number of revolutions to 0, and save/close the coast. Voila! You're done! Be sure to save the mission plan at this point in time, too.

Lessons Learned

A variety of important skills, tips, and tricks have been learned in this section. To summarize, they are:

1. Parking orbits should be in the same orbital plane as the hyperbolic departure orbit in order to minimize the required departure delta-v.
2. Allow plenty of room in your pre-departure burn coast (here, -180 deg to 360 deg) to give the optimizer room to work within.
3. Removing the radial component of the departure burn can eliminate wasteful delta-v.
4. Most of your departure burn delta-v should be in the prograde direction, so set smaller bounds on the normal component of the burn. But allow enough room for the optimizer to adjust things as needed!
5. Make sure that your initial guess for the prograde component of the burn is enough to eject the spacecraft from Kerbin orbit. The post burn eccentricity should be greater than 1.0.

6. Optimizing a maneuver to achieve a particular orbit at your flyby destination is a multi-step process. You cannot do it in one step.
7. Your first goal with the optimizer should be to simply hit the target body as directly as possible. Your next goal should be to achieve the desired orbital plane (inclination, RAAN, argument of perapsis) at the correct time (UT). Finally, your third optimization run can deal with achieving the required SMA and eccentricity.
8. When your flyby altitude is very close to the atmosphere, consider setting a radius of periapsis constraint instead to prevent inadvertent re-entry.
9. If you have trouble getting your solution to converge with an SMA constraint, consider using a radius of periapsis constraint instead.
10. If you have trouble getting your solution to converge with one objective function, try using another objective function.

Tutorial Part 4 – Eve Powered Flyby, Eve Departure, and Jool Arrival

In this phase of the mission, we will create our powered Eve flyby maneuver, create coasts to Jool's periapsis, and use the optimizer to target the burn such that we achieve the desired inbound hyperbolic orbit at Jool.

Creating the Powered Flyby

A powered flyby is merely a normal flyby maneuver (also known as a gravity assist maneuver) during which the spacecraft engines are fired to provide more “boost.” The engines may be fired at any point during the flyby, but typically such maneuvers occur near periapsis. In MFMS, all powered flybys are designed to occur at periapsis.

In the main Mission Architect window, you should note that the Final Spacecraft State has a true anomaly of very nearly zero. This is desirable, as the maneuver described in the MFMS output occurs at periapsis.

Insert a Delta-V Maneuver into the mission script following the “Coast to Eve Peri” event. The familiar Create DV Maneuver dialog box should appear. Recall from the MFMS output the parameters of this maneuver, shown in Table 6.

Table 6: Burn Information to Depart Eve

Burn Information to Depart Eve	

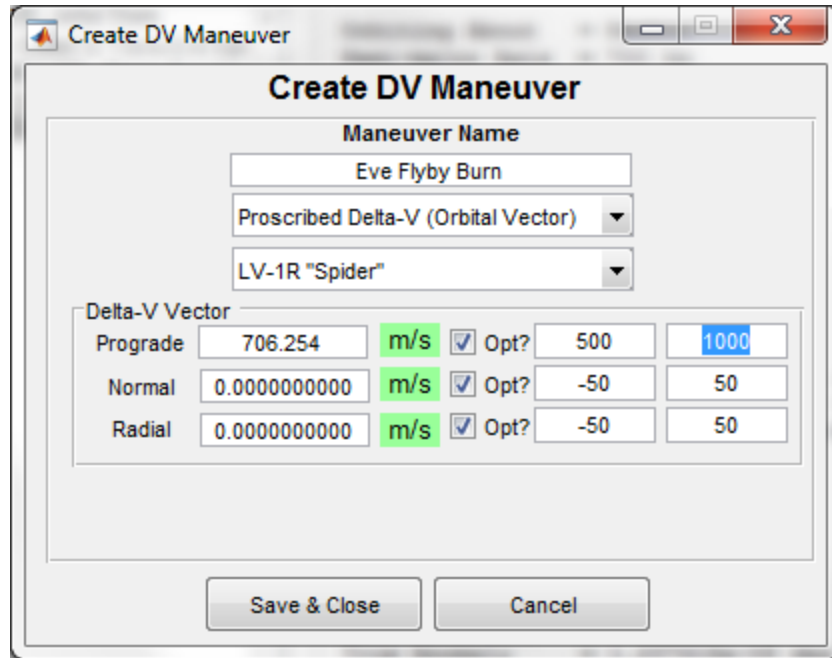
Total Delta-V =	0.706 km/s
Prograde Delta-V =	706.254 m/s
Orbit Normal Delta-V =	-0.000 m/s
Radial Delta-V =	-0.000 m/s

Burn True Anomaly =	0.000 deg

Unlike the Kerbin departure burn, the maneuver information here is most likely reasonably accurate. It makes a good “first guess” as to the maneuver parameters. Name the burn “Eve Flyby Burn” and ensure that the maneuver type is “Proscribed Delta-V (Orbital Vector)”. Populate the delta-v vector with the parameters in Table 6.

As usual, selecting the delta-v vector upper and lower bounds is tricky. We do know that we would like to minimize the amount of radial and normal delta-v, as these components are inefficient and, ideally, should not be required. Let's set the upper and lower bounds on these components to ± 50 m/s. The prograde component should be given more flexibility as it is the primary driver of our next Sun-centered transfer orbit to Jool. Let's use 500 m/s on the lower bound and 1000 m/s on the upper bound. As usual, these bounds are both science and art and the values selected here are mostly derived from prior experience with Mission Architect.

When you've finished entering the burn parameters, save and close the dialog box. Your burn should look like that shown in Figure 18.



Create DV Maneuver

Maneuver Name
 Eve Flyby Burn

Proscribed Delta-V (Orbital Vector)
 LV-1R "Spider"

Delta-V Vector

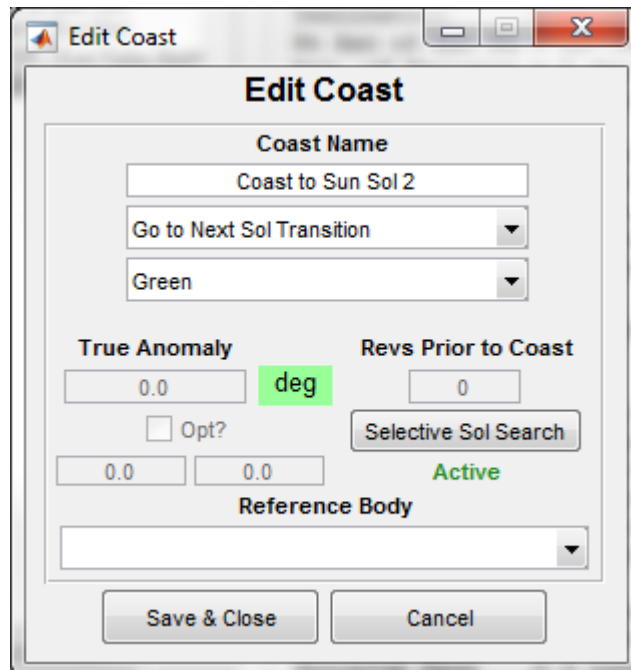
	Value	Unit	Opt?	Value	Value
Prograde	706.254	m/s	<input checked="" type="checkbox"/>	500	1000
Normal	0.0000000000	m/s	<input checked="" type="checkbox"/>	-50	50
Radial	0.0000000000	m/s	<input checked="" type="checkbox"/>	-50	50

Save & Close Cancel

Figure 18: Eve Flyby Burn

Next, we need to create another new two coast events, as we did in order to depart Kerbin.

Create the first Coast event to go to the next SOI transition, here from Eve to the Sun. Name the Coast "Coast to Sun SOI 2" and use Green as the color. Save and close the coast when you've entered all the required information. It should look as shown in Figure 19 below.



Edit Coast

Coast Name
 Coast to Sun Sol 2

Go to Next Sol Transition

Green

True Anomaly
 0.0 deg

Revs Prior to Coast
 0

☐ Opt?

0.0 0.0 **Active**

Reference Body

Save & Close Cancel

Figure 19: Coast to Sun SOI 2

Finally, create the coast to Jool's periapsis in a manner similar to that used to create the coast to Eve's periapsis. Name the Coast "Coast to Jool Peri" and select the coast type "Go to Periapsis." Use Cyan as the coast color. Here again, as with the coast to Eve, the number of revolutions should be set to 1. We will adjust it back to zero later after successfully targeting Jool. Set the Reference Body to Jool (this is **very important**). Finally, push the Selective SOI Search button; select Sun, Jool, Laythe, Vall, Bop, Tylo, and Pol.

Once you've completed entering the coast parameters, save and close the dialog box. Your coast should look identical to Figure 20 below.

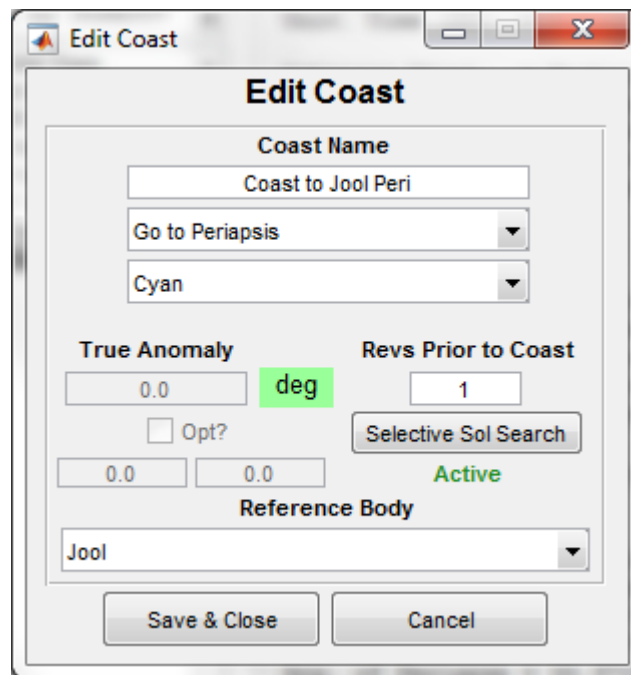


Figure 20: Coast to Jool Peri

If your own copy of the mission plan in Mission Architect has been keeping pace with this tutorial's, you will notice that upon creating the coast to Jool's SOI, your trajectory does in fact make it to Jool's SOI without optimizing. This is actually a fairly frequent occurrence as Jool's SOI is quite large and therefore a pretty easy target. So much the better for us, though.

We're now ready to optimize our burn plan to achieve our target inbound Jool orbit.

Optimizing the Eve Departure Burn

In this section, we will step through the process of optimizing the Eve departure burn in order to target our desired orbit at Jool. In order to set up for this, we should disable the optimization flags on our Kerbin departure burn and coast parameters. Just uncheck the "Opt?" box in the coast event and the three "Opt?" boxes in the DV maneuver event. It is good practice to only optimize one portion of the trajectory at once. This is because very small changes in burn parameters in a prior segment can have very large impacts on the downstream trajectory, and the optimizer will have a hard time juggling these

large impact changes with the smaller impact changes that occur when varying parameters in the current mission phase.

When you are finished doing this, there should be no asterisk (*) next to Event 2 and Event 3 in the mission script list. The only item with an asterisk should be Event 6, the Eve Flyby Burn. Your Mission Architect window should look as shown in Figure 21.

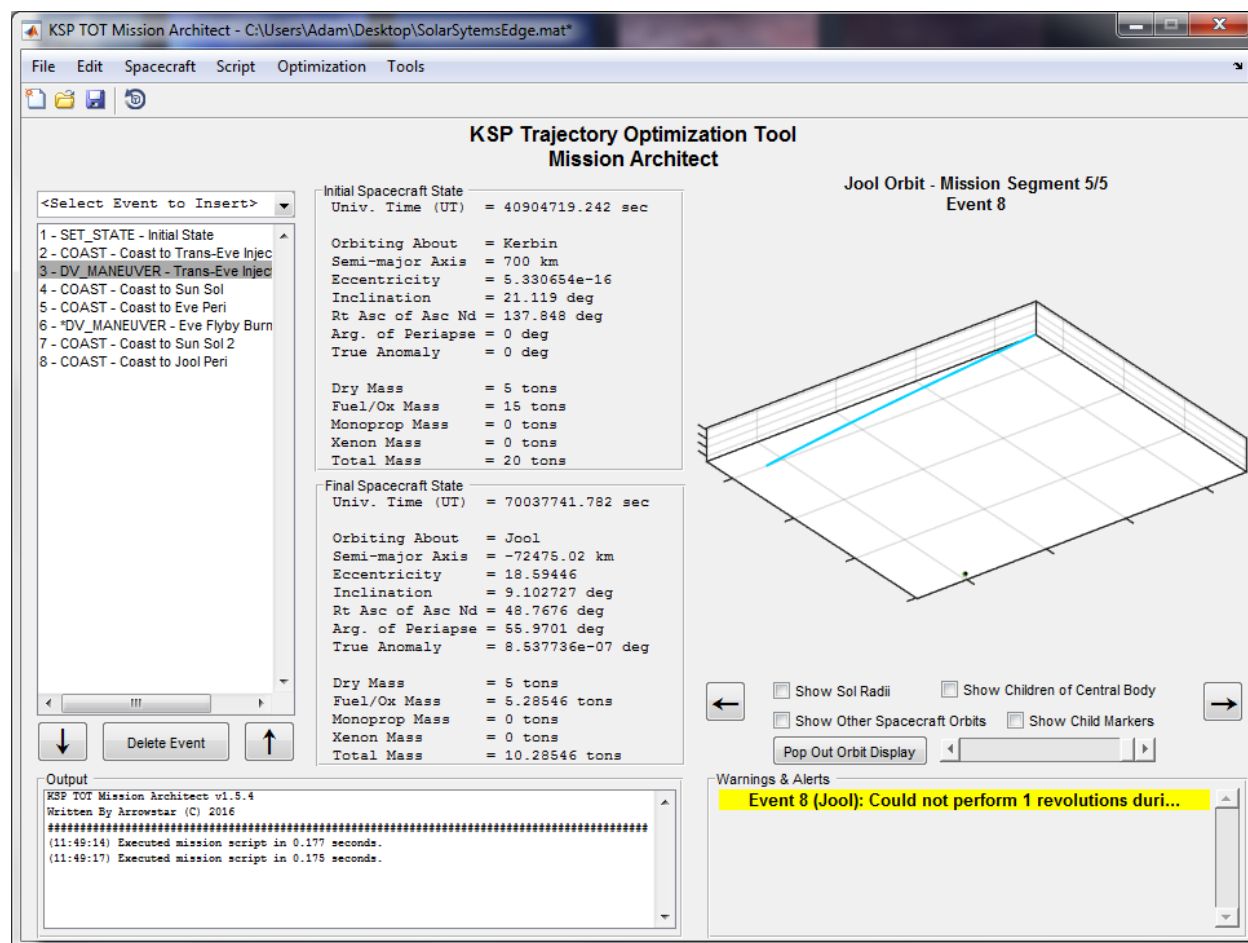


Figure 21: Mission Architect Window Prior to Eve Flyby Burn Optimization

Optimizing the Eve Flyby Burn – Round 1

Go ahead and open the Mission Optimizer. As with Round 1 of the Kerbin departure burn optimization, the goal of this attempt will be to target the correct inbound hyperbolic orbit plane. We also want to make sure that our final trajectory stays at Jool.

As an aside, notice that when you open the optimizer, all of your previously defined constraints are still present from the other set of optimizer runs we did. So long as you have no active variables prior to the events associated with these constraints (and you shouldn't have any like this now), you may leave these constraints as they are. They will not impact the optimization run. In fact, it is good practice to leave prior constraints associated with the mission plan, even if you do not intend to use them again. If, by chance, you must go back and modify an earlier segment of your mission plan, you will not have to

recreate the events in the future. It also provides a form of documentation as to how you set up your optimization runs.

To ensure our final trajectory stays at Jool, first create a Central Body constraint on Event 8 with Jool as the Celestial Body.

To set the orbit plane constraints, first recall our ideal inbound Jool trajectory, shown in Table 7.

Table 7: Inbound Hyperbolic Flyby Orbit to Jool

Inbound Hyperbolic Flyby Orbit to Jool	

Semi-major Axis =	-67642.588 km
Eccentricity =	1.0917
Inclination =	5.413 deg
Right Ascension of AN =	117.899 deg
Argument of Periapse =	54.137 deg
Periapse Radius =	6200.001 km

Create an inclination constraint, a RAAN constraint, and an argument of periapsis constraint. The upper and lower bounds of each constraint should be that of the respective value in Table 7, ± 0.25 degree. Ensure that Jool is set as the Celestial Body for each of these three constraints as well.

You will notice that the tolerances on these constraints are tighter than the previous Eve arrival trajectory. This is because our next target from Jool is Plock, a body with a small sphere of influence (as compared to that of Eve or Jool, which have large spheres of influence). Because of the small size of Plock's SOI, it is better to try and be more exact on the inbound trajectory.

After you've created these constraints, ensure that the objective function is still set to "No Optimization." We merely want to satisfy the constraints, without driving the solution towards Jool quite yet. We will handle that manually in the next step.

When you are ready, push the Optimize button and wait for the optimizer to compete. It should easily home in on the constraint values.

After this run, your Mission Architect window should look very similar to what is shown in Figure 22 below.

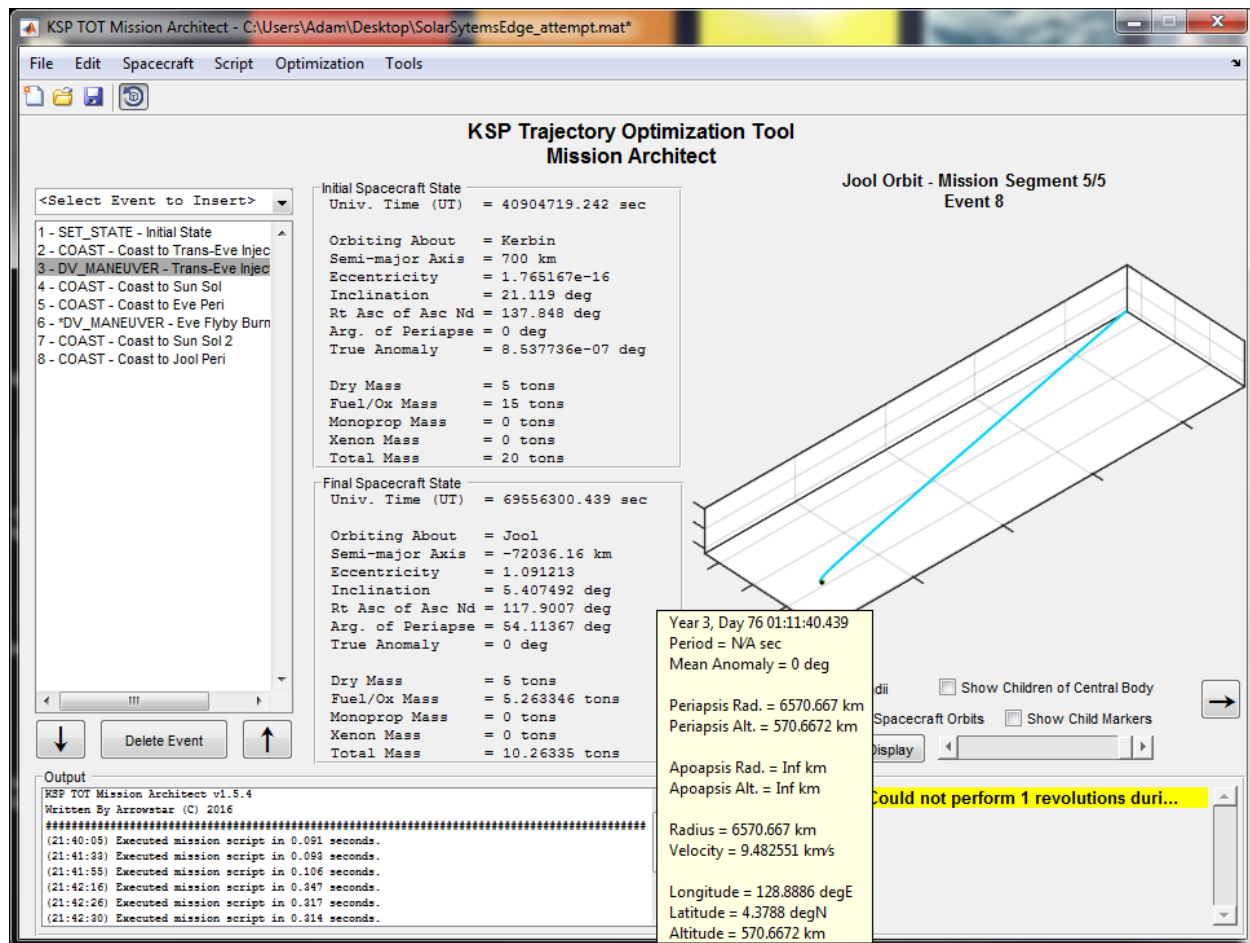


Figure 22: Mission Architect after Eve Flyby Burn Optimization - Round 1

Optimizing the Eve Flyby Burn – Round 2

In this round, our optimization goal will be to target the periapsis radius. A quick warning up front: in my experience here, achieving this target may be tricky and we may have to settle for constraints which are not fully satisfied (though they should be close!).

Open up the Mission Optimizer once more. Create a radius of periapsis constraint. Notice that the radius of Jool itself is 6000 km and the atmospheric height is 200 km. This means that our target Jool periapsis is right along the edge of the atmosphere, as with our Eve flight. This is not surprising, as the closer you get to a celestial body during a flyby, the more boost you get from that body.

The new radius of periapsis constraint should have a lower bound of 6200 km and an upper bound of 6215 km. Save and close the constraint dialog box when you've entered the parameters.

When you are ready, re-optimize the mission with the new constraint. Be patient here, as this optimization run may take some time, perhaps up to 50-100 iterations.

When the optimizer finishes, hopefully with either a 0 or very low constraint violation value, your Mission Architect window should look like the following in Figure 23.

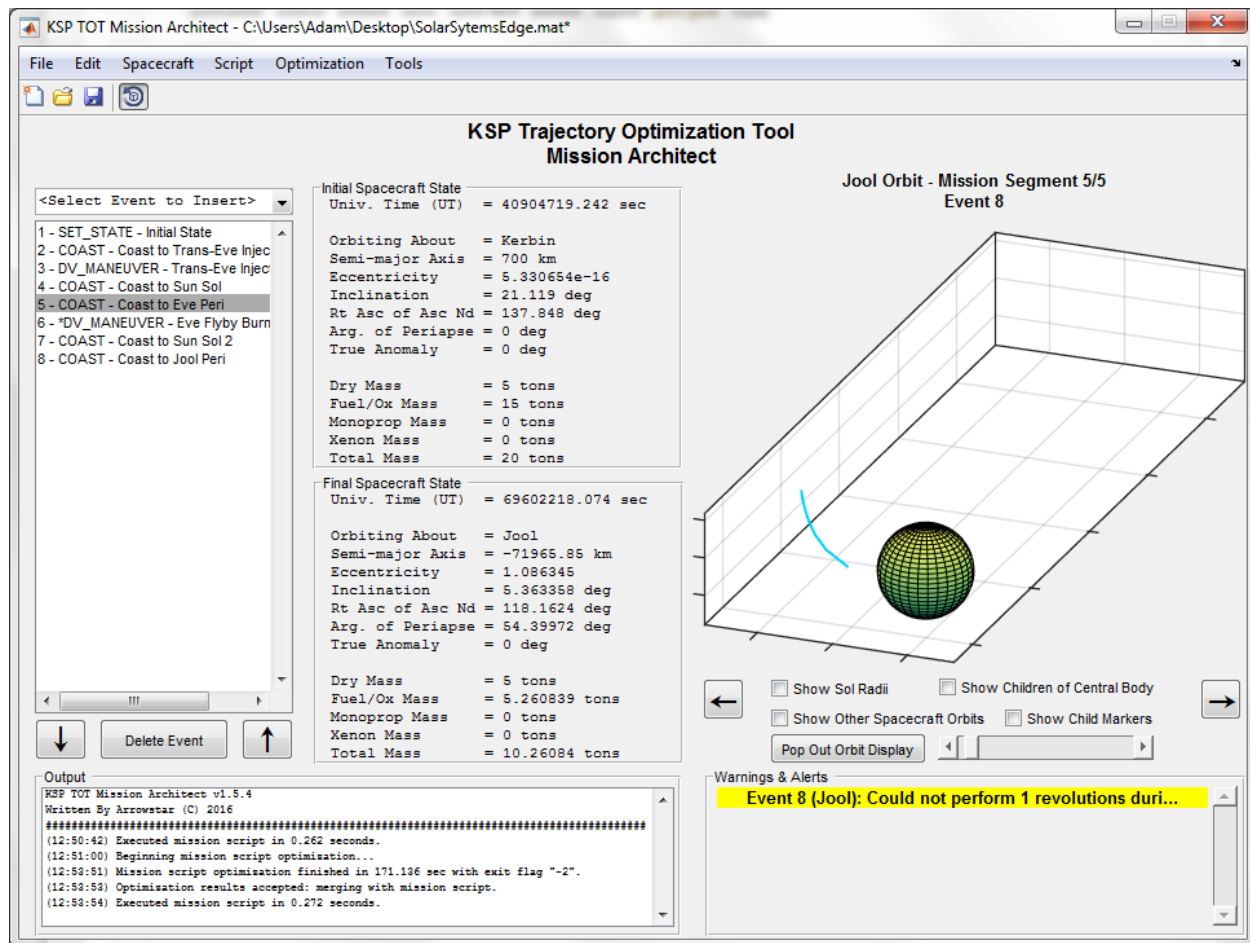


Figure 23: Mission Architect after Eve Flyby Burn Optimization - Round 2

Now, we have an interesting decision to make here. It turns out that a fair bit of trial and error on my end demonstrates that the eccentricity target of 1.0917 isn't really achievable without adding an extra degree of freedom in the form of a moveable burn true anomaly at Eve. Since I prefer that we don't add this now, and the eccentricity constraint is hard to achieve, we will leave the mission plan as-is after this optimization run.

The added side effect of this adjustment to the mission design is that we will now need to create a powered flyby at Jool where before we simply had an unpowered flyby. The next section will cover this in detail.

To conclude this section, open up the final "Coast to Jool Peri" event and set the number of revolutions to 0 in order to remove the warning. Then save your mission plan.

Lessons Learned

A variety of important skills, tips, and tricks have been learned in this section. To summarize, they are:

1. Powered flyby maneuvers are most efficient when performed at periapsis.

2. Always work on only one section of the mission at once, and disable optimization flags on variables in other sections.
3. Set the number of post-SOI change revolutions to 1 when targeting another body to give the optimizer as much orbit path to work with.
4. Use of the Selective SOI Search feature can speed up optimization and trajectory propagation.
5. Jool's SOI is large and it is typically a very easy target to hit, even without significant amounts of prior trajectory optimization.

Tutorial Part 5 – Powered Jool Flyby, Jool Departure and Plock Arrival

In this phase of the mission, we will create our powered Jool flyby maneuver, create coasts to Plock's periapsis, and use the optimizer to target the burn such that we achieve the desired inbound hyperbolic orbit at Plock.

Creating the Powered Flyby

You'll recall that the problem we had in the last section revolved around not having a movable burn at Eve in order to better target the Jool pass. Since we have a very tight target for our Jool powered flyby to achieve (namely, arrival at Plock), we likely cannot afford this "mistake" again. Thus, we are going to alter the Coast to Jool Peri event first, before we begin, in order to allow our optimizer to move the burn slightly.

Open the Coast to Jool Peri event. Change the event type to "Go to True Anomaly." Notice that the True Anomaly field reads 0.00 degrees. If it does not, set it to 0.00 degrees.

Check the "Opt?" box to tell the optimizer to adjust this parameter. The bounds we use here should be fairly tight: recall that we want to take advantage of Jool's gravity as much as possible first, so our burn should be very close to the periapsis anyway. Let's use -5 degrees for the lower bound and 5 degrees for the upper bound.

When you've completed this, save and close the dialog box. Your coast should look identical to Figure 24.

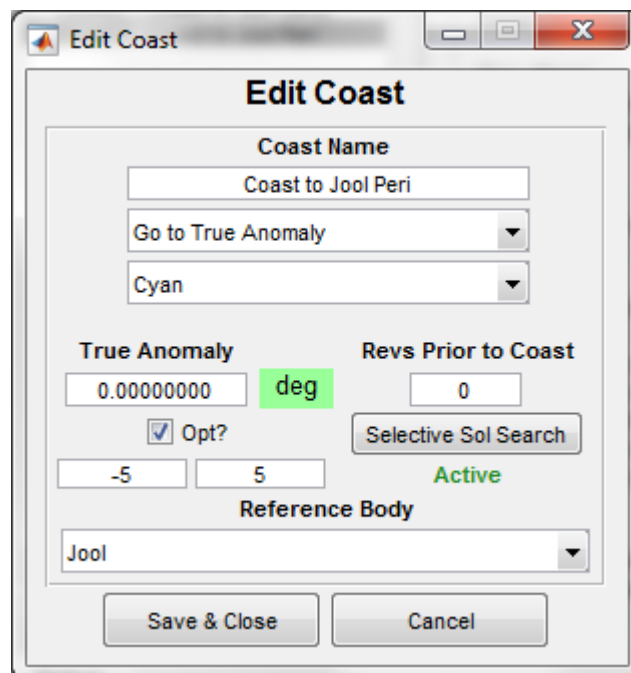


Figure 24: Coast to Jool Peri Adjustment

Next, let's create the powered flyby itself. Add a DV Maneuver event after the Coast to Jool Peri event. Because our Jool orbit is slightly off from the mission plan out of MFMS and the maneuver at Jool in that plan was effectively nothing (only ~8 m/s), we don't really have much information again on what should be required to target Plock successfully. We know that some prograde delta-v may be necessary, and again the radial and normal components shouldn't be too large.

Based on this, set all components of the delta-v vector at 0 m/s. The bounds on components can be ± 200 m/s, which should be fairly generous.

Name the burn "Jool Flyby Burn". When you are done, save and close the dialog box. Your completed burn should look like Figure 25.

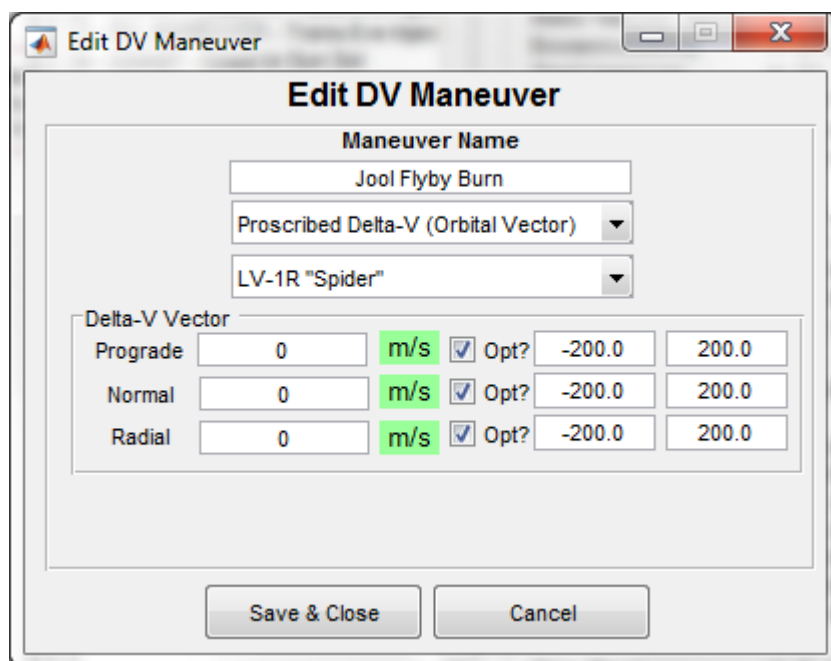


Figure 25: Jool Powered Flyby

Next, we need to add the Coast to the next SOI transition and the coast to Plock's periapsis. Go ahead and add these. The former should be called "Coast to Sun SOI 3" and use the color Blue. The latter should be called "Coast to Plock Peri 1" and use the color Magenta. Make sure the reference body is set to Plock on this coast. Also ensure that you set the Selective SOI Search to only use the Sun, Plock, and Karen. The "revs prior to coast" parameter in this burn should be set to 1 for now, as normal.

When you are done, the two coasts should look like the following shown in Figure 26.

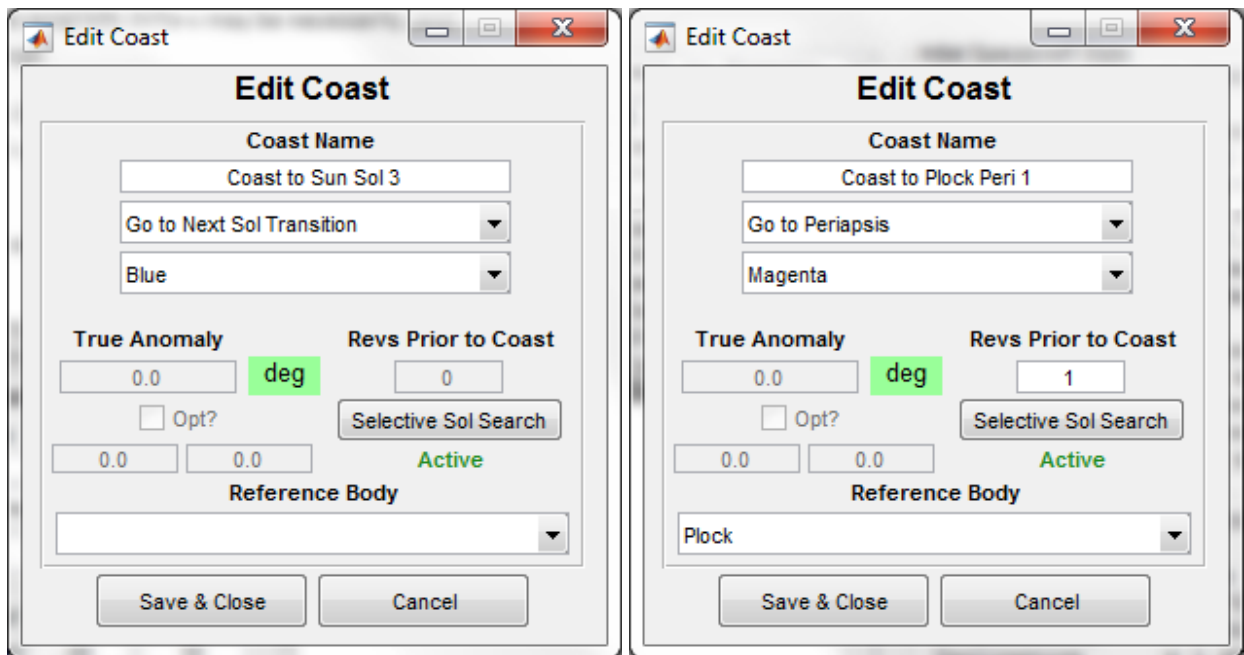


Figure 26: Coast to Sun SOI 3 and Coast to Plock Peri 1

Finally, remove the optimization flags on Event 6, the Eve Flyby Burn. We should not be modifying the Eve burn when we are primarily aiming to adjust the burn at Jool.

When you've completed this section, the Mission Architect window should look like Figure 27.

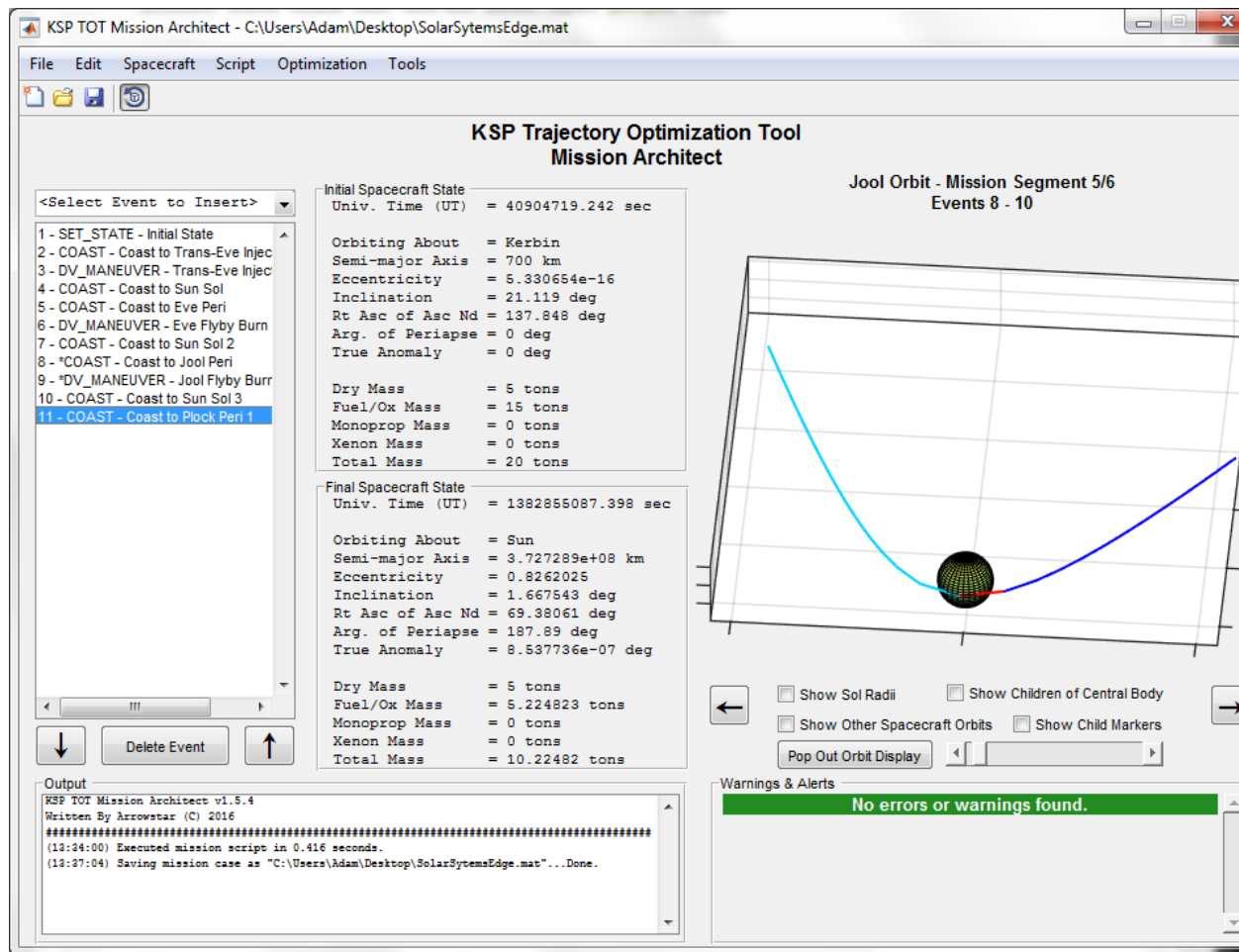


Figure 27: Mission Architect window prior to optimizing Jool Flyby Burn

Optimizing the Jool Flyby Burn – Round 1

In this segment, our primary goal is to simply get our trajectory into the Plock system.

Open up the Mission Optimizer. If you had any unsatisfied constraints from our last runs, you'll need to disable those constraints before the optimizer will really run well. At this time, go ahead and open up each constraint in the Active Constraint list and uncheck the two checkboxes. Doing so will deactivate the constraint and tell the optimizer to ignore it. You may skip the Central Body constraints, as these can't be deactivated in this manner. However, as those are already satisfied, there is little harm to leaving them active.

Now we need to target Plock. Set the objective function to Minimize Distance to Body, the Applicable Event to Event 11, and the Body to Plock. When you're ready to go, push the Optimize Mission button. Your Mission Optimizer window should look like Figure 28 below.

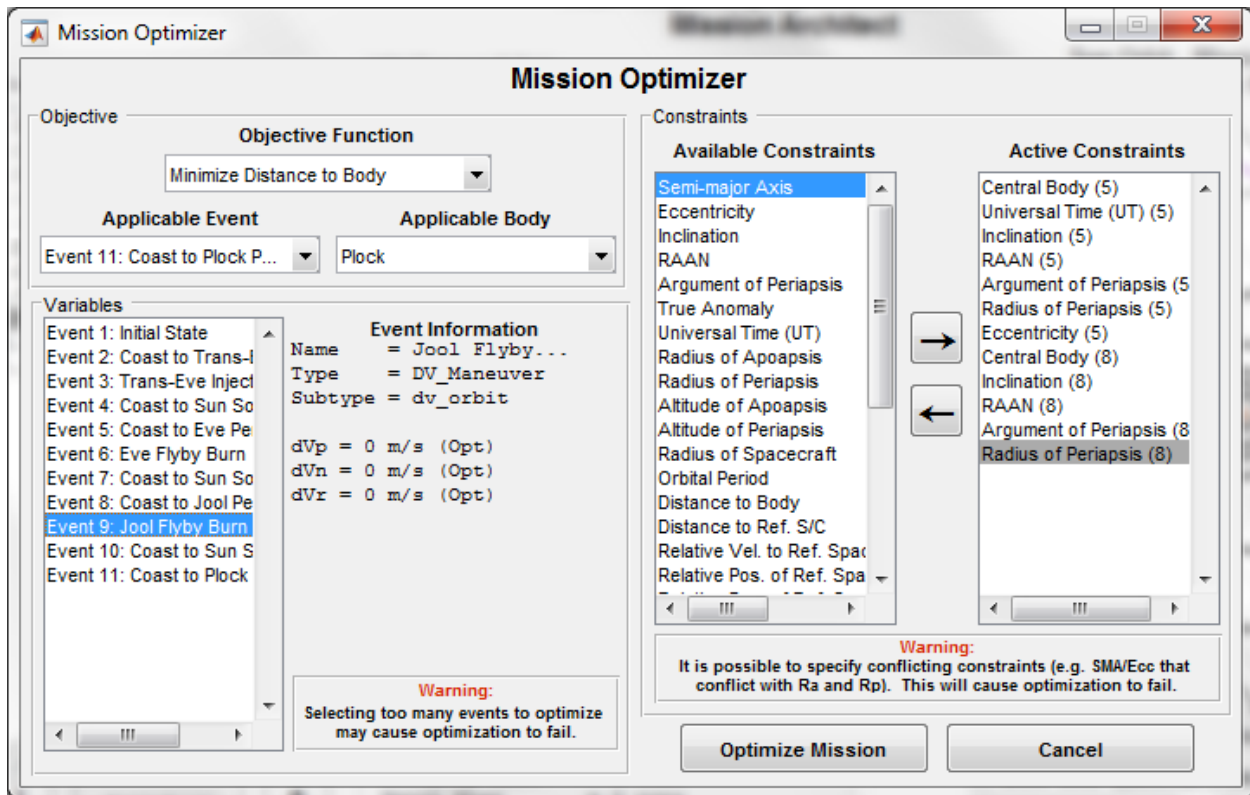


Figure 28: Mission Optimizer window prior to optimizing Jool Flyby Burn - Round 1

Don't be surprised if this optimization run takes some time to complete. The Plock system is fairly small and has a small sphere of influence. It may take a number of iterations for the optimizer to really home in on Plock.

When the optimization run completes, you should end up with a trajectory that takes you into the Plock system and something similar to what is shown in Figure 29.

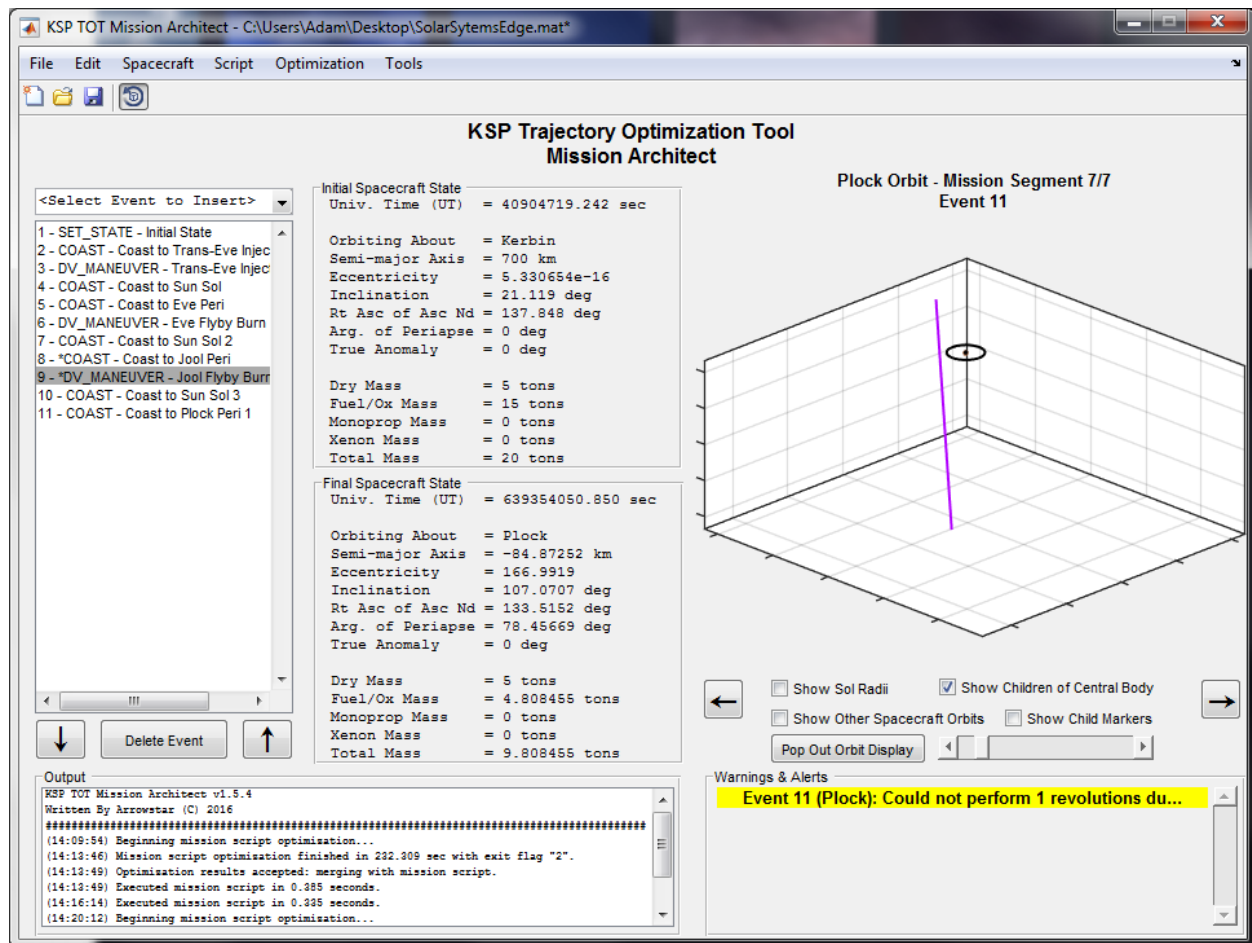


Figure 29: Mission Architect after Jool Flyby Burn Optimization - Round 1

Optimizing the Jool Flyby Burn – Round 2

The goal of this segment will be, as usual, to place our spacecraft into an orbit at Plock that matches the target orbital plane. In this case, that orbital plane is that of our rendezvous spacecraft, the Plock Orbiter 1. Recall that this vehicle is in a circular, equatorial orbit around Plock. Therefore, we would like to simply minimize the amount of inclination we have remaining when our spacecraft reaches Plock's periapsis.

Open up the Mission Optimizer. Leave the objective function at "Minimize Distance to Body". Create a Central Body constraint on Event 11 and set the Celestial Body to Plock. Now create an inclination constraint. Set the lower bound to 0 degrees and the upper bound to 30 degrees. The idea here is to provide some room for the optimizer to work within while still trying to guide the inclination downwards towards a reasonable value. If you'd like, you can also try a "Minimize Inclination" objective function here, but it's not critical.

Once you've added these constraints, go ahead and optimize the mission plan. Your Mission Optimizer window should look like that shown in Figure 30.

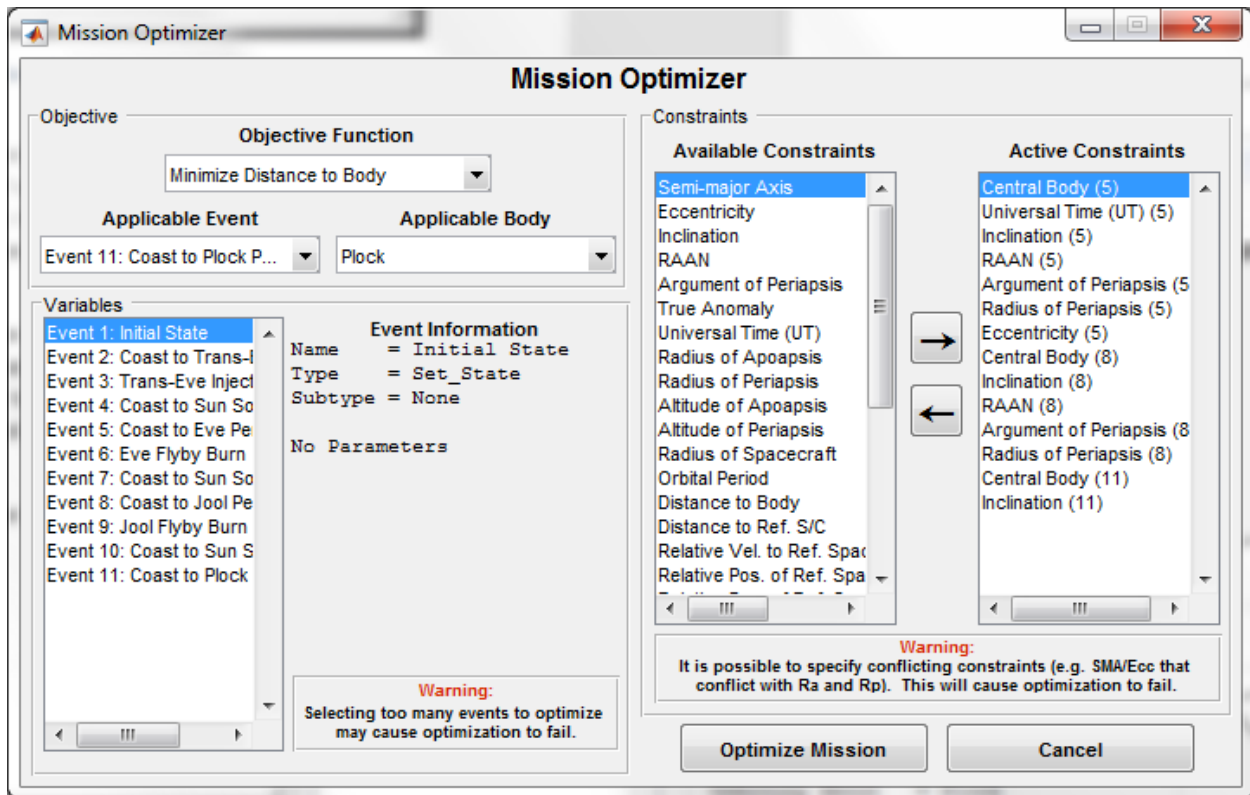


Figure 30: Mission Optimizer window prior to optimizing Jool Flyby Burn - Round 2

After the optimization completes, your Mission Architect window should look like that shown below in Figure 31. In particular, pay attention to the inclination, eccentricity, and argument of periapsis values in the figure.

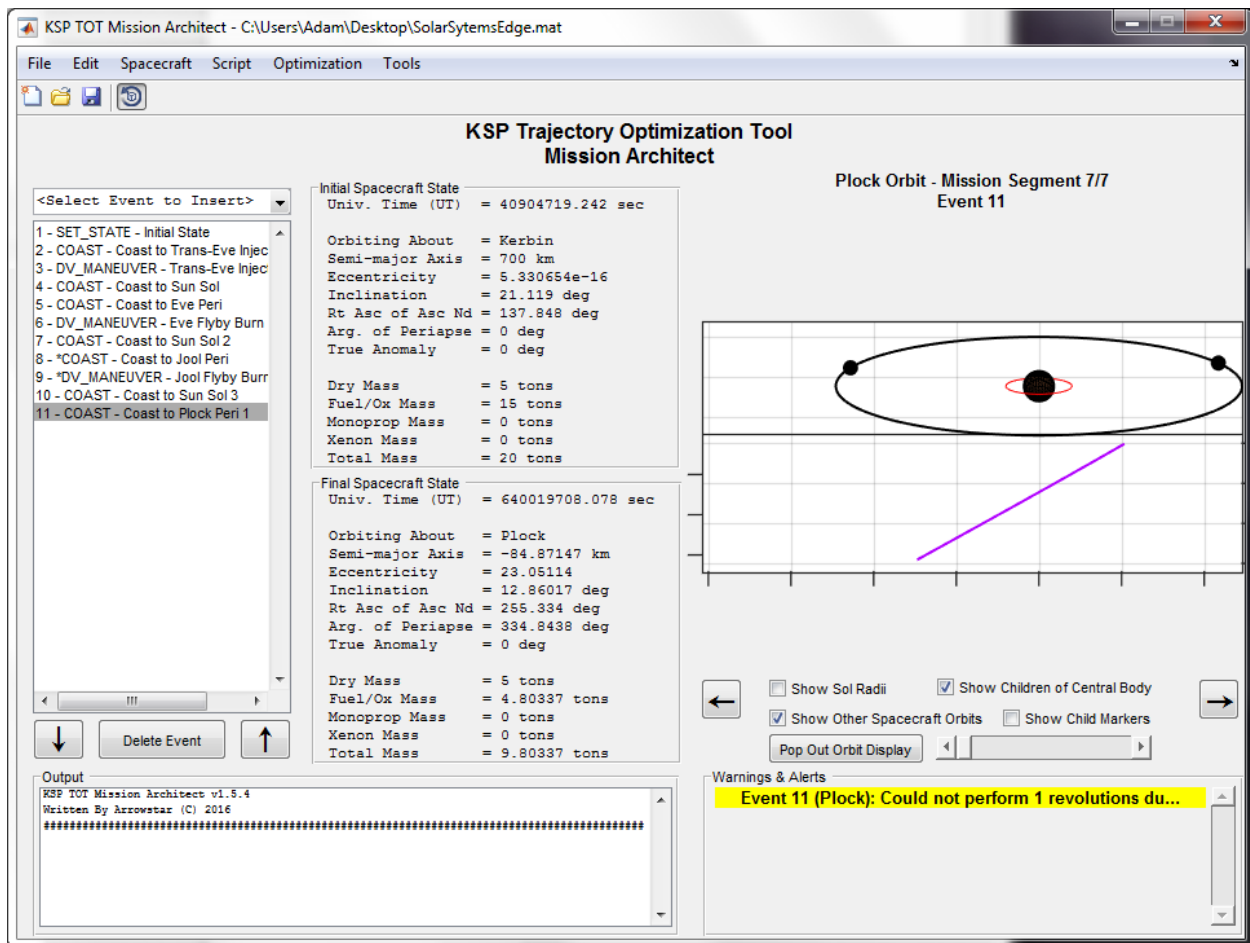


Figure 31: Mission Architect after Jool Flyby Burn Optimization - Round 2

This concludes the Jool flyby burn optimization. The spacecraft trajectory is successfully approaching Plock on an inbound trajectory which should be capable of successfully rendezvousing with the Plock Orbiter spacecraft. As usual, save your mission plan here.

Lessons Learned

A variety of important skills, tips, and tricks have been learned in this section. To summarize, they are:

1. Problems with achieving your target trajectory in one mission section can result in necessary changes to the trajectory or burn plans in a downstream section.
2. Unchecking both checkboxes in the Mission Architect Optimizer constraint windows deactivates that constraint.

Tutorial Part 6 – Plock Arrival and Rendezvous Activities

In this section, we will coast deep into the sphere of influence of Plock and rendezvous with the Plock Orbiter 1 spacecraft.

Setting Up the Rendezvous

It is possible to plan rendezvous maneuvers fully in Mission Architect. However, KSPTOT actually includes an application designed to perform these calculations quickly and accurately, the Rendezvous Maneuver Sequencer (RMS). In this next segment, we will discuss using the RMS tool to compute the two burn maneuvers we need to rendezvous with the target spacecraft. A completed example burn plan will be presented.

Setting Up the Rendezvous Maneuver Planner

Navigate back to the main KSPTOT UI, the one that contains the porkchop plotter. Use the menu “Tools -> Maneuver Planning -> Rendezvous Maneuver Planner” to open up the Rendezvous Maneuver Planner.

Update the “Orbiting About” body in the upper left to match our current body in Mission Architect, Plock.

In the “Final Orbit” segment in the lower left, enter the parameters for the Plock Orbiter 1 spacecraft that were described in the first part of this tutorial.

Next, we will populate the “Initial Orbit” segment in the upper left. To do so, navigate back to Mission Architect. Right click on the “Final Spacecraft State” box in the middle of the UI and click “Copy Orbit to Clipboard.” This places the orbit parameters on an internal KSPTOT orbit clipboard that can then be pasted around the application. Navigate back to RMS. Right click anywhere in the “Initial Orbit” area and select “Paste Orbit from Clipboard.”

Finally, we need to set the “Initial Epoch” parameter at the bottom of the UI. Copy the “Epoch” from the “Initial Orbit” area and paste it into the “Initial Epoch” parameter.

When you are finished, your UI should look similar to Figure 32. The “Initial Orbit” area may not match exactly.

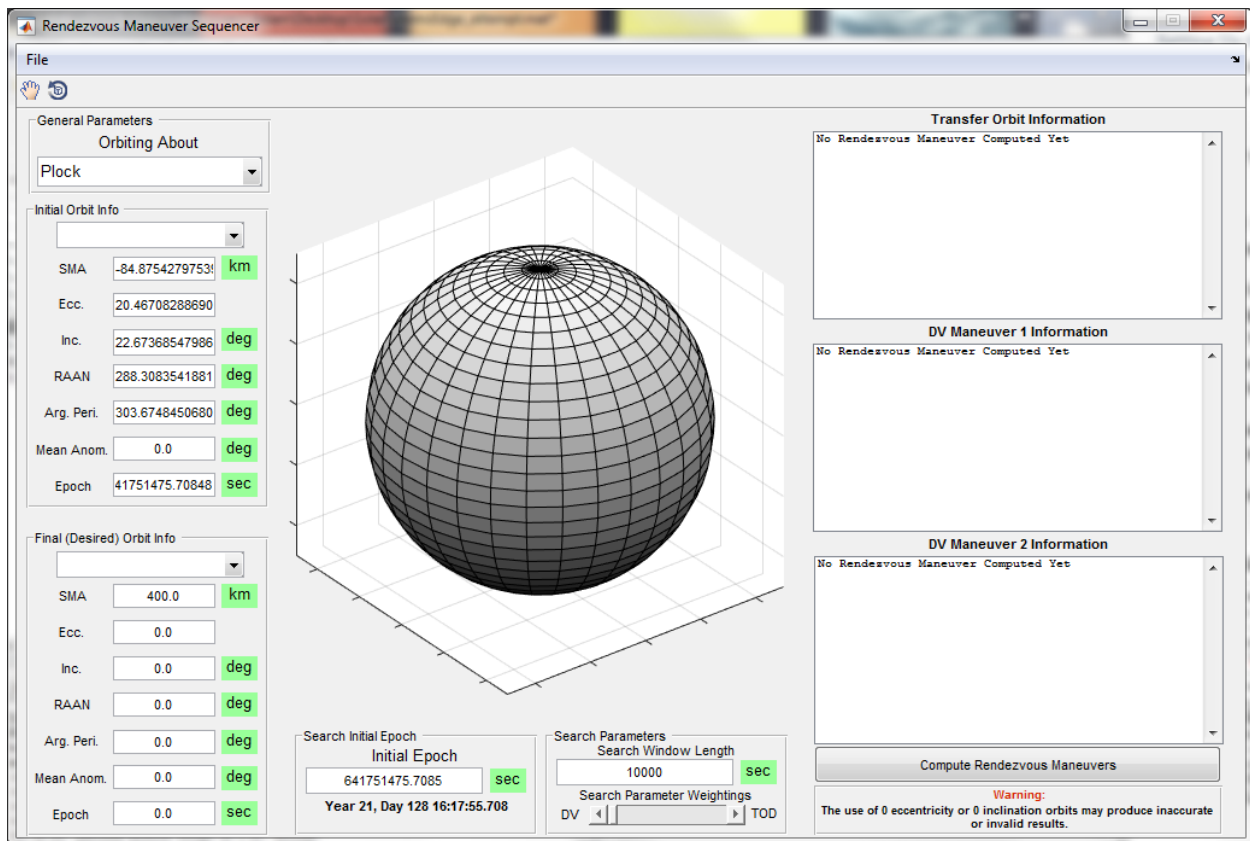


Figure 32: Rendezvous Maneuver Sequencer Setup

Now push the "Compute Rendezvous Maneuvers" button at the bottom right and wait for the application to finish. When it has, the RMS user interface should populate with information similar to that shown below in Figure 33.

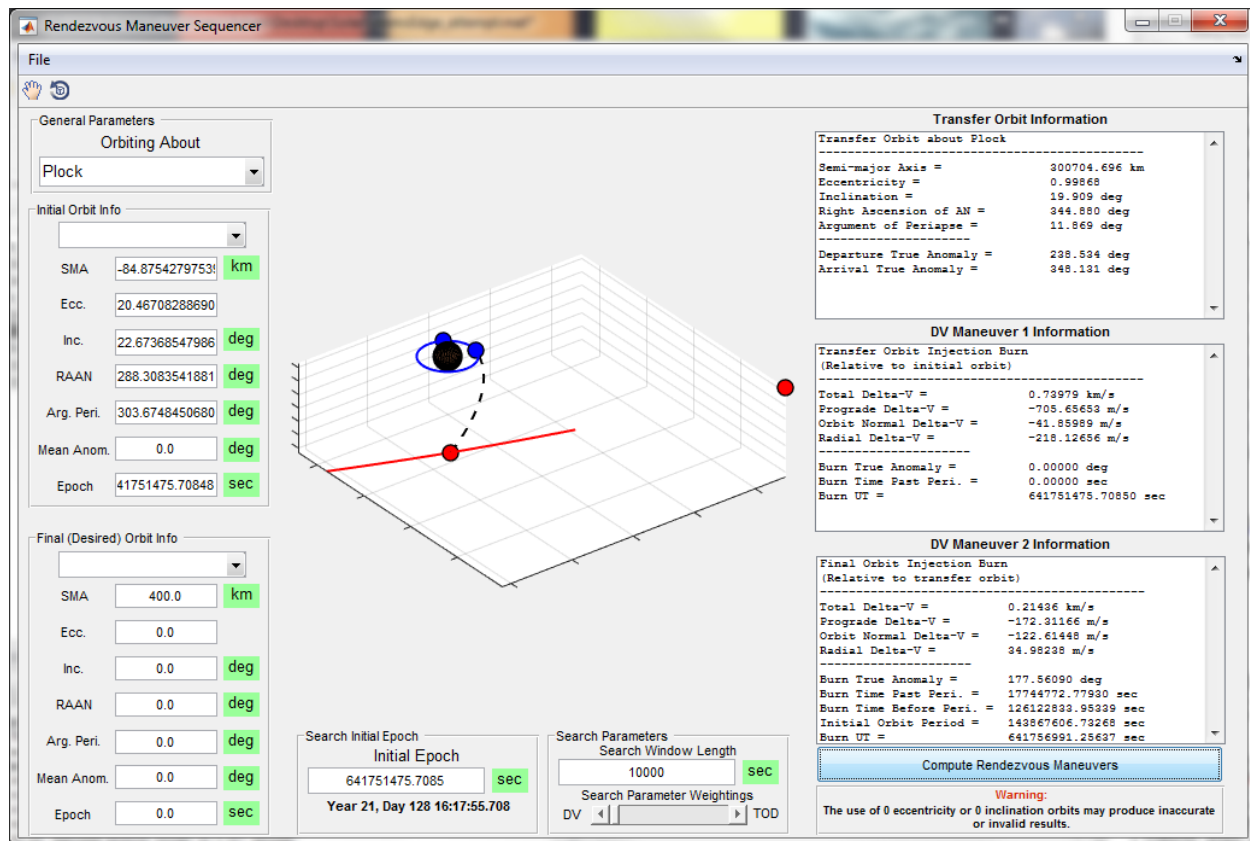


Figure 33: Rendezvous Maneuver Sequencer After Calculations

A cleaner version of the information that RMS provided me is shown below. In the following section, you will want to use the information RMS provided you. Table 8 should only be taken as an example and it will likely not work for your specific case.

Table 8: Rendezvous Maneuver Sequencer Output

Transfer Orbit about Plock	

Semi-major Axis =	300704.696 km
Eccentricity =	0.99868
Inclination =	19.909 deg
Right Ascension of AN =	344.880 deg
Argument of Periapse =	11.869 deg

Departure True Anomaly =	238.534 deg
Arrival True Anomaly =	348.131 deg

Transfer Orbit Injection Burn (Relative to initial orbit)	Final Orbit Injection Burn (Relative to transfer orbit)

Total Delta-V =	0.73979 km/s
Prograde Delta-V =	-705.65653 m/s
Orbit Normal Delta-V =	-41.85989 m/s
Radial Delta-V =	-218.12656 m/s

Burn True Anomaly =	0.00000 deg
Burn Time Past Peri. =	0.00000 sec
Burn UT =	641751475.70850 sec

Total Delta-V =	0.21436 km/s
Prograde Delta-V =	-172.31166 m/s
Orbit Normal Delta-V =	-122.61448 m/s
Radial Delta-V =	34.98238 m/s

Burn True Anomaly =	177.56090 deg
Burn Time Past Peri. =	17744772.77930 sec
Burn Time Before Peri. =	126122833.95339 sec
Initial Orbit Period =	143867606.73268 sec
Burn UT =	641756991.25637 sec

Targeting the Transfer Orbit

In this section, I am going to reference the results from my RMS run in the previous section. You should replace my results with yours at each step.

According to Table 8, a burn at periapsis of 739.79 m/s will alter the trajectory of our spacecraft and place it on a path towards meeting with the Plock Orbiter. This is called the “Transfer Orbit” and this burn is called the “Transfer Orbit Injection Burn.” Let’s model that burn in Mission Architect now.

Because the burn takes place at periapsis and our current coast event into Plock terminates at periapsis, no additional coast events are required. Instead, we can simply add the burn. Insert a DV Maneuver as Event 12. Call the burn “Rendezvous Maneuver 1” and ensure that the burn type is “Proscribed Delta-V (Orbital Vector).” Now add the prograde, normal, and radial components from the “Transfer Orbit Injection Burn” portion from your RMS results to their respective DV components in the Create DV Maneuver dialog box. For now, we’ll leave the optimization checkboxes checked and leave the default bounds of ± 1000 m/s as they are.

Save and close the dialog box. The Mission Architect window after adding this maneuver should look similar to Figure 34 below.

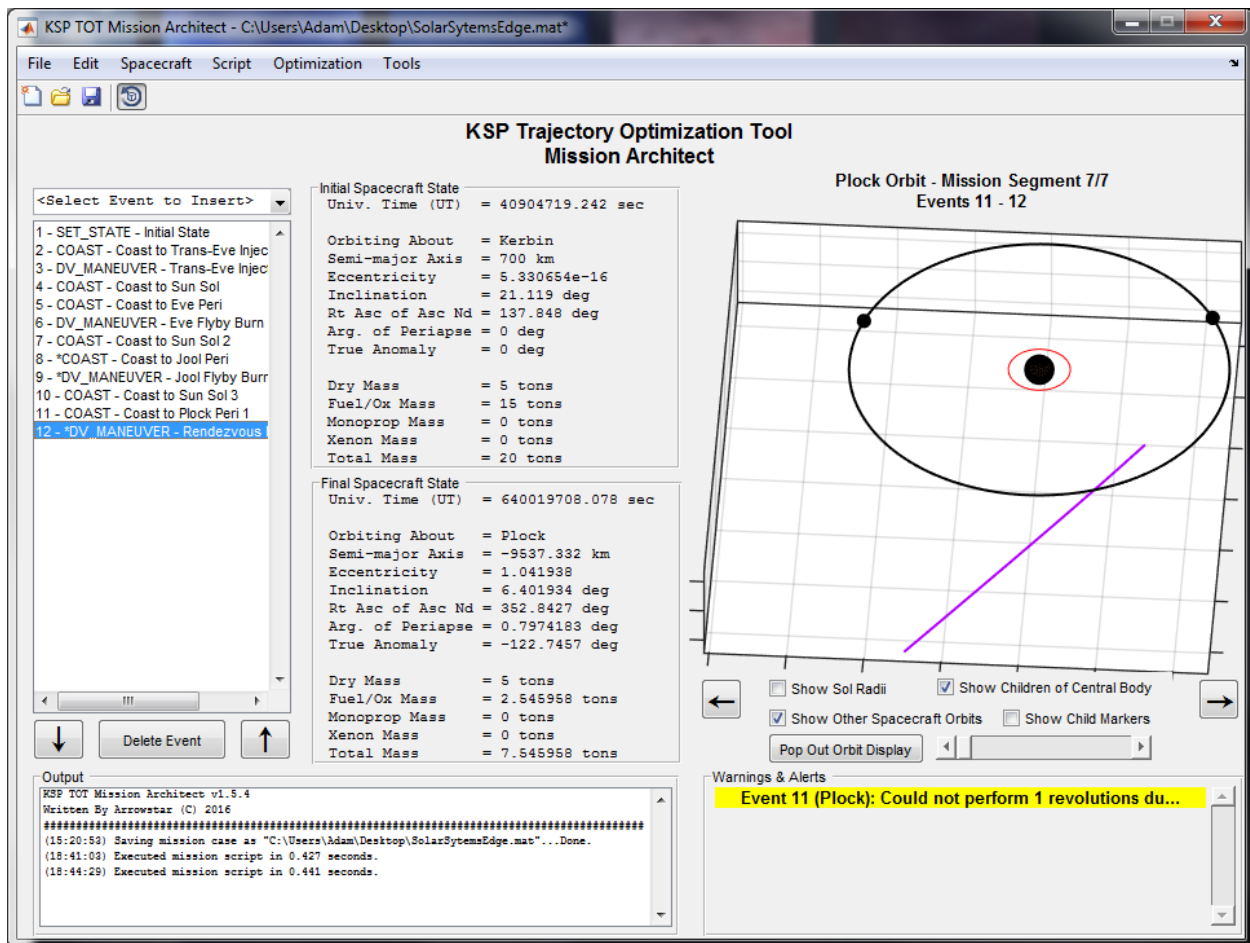


Figure 34: Mission Architect after Adding Rendezvous Maneuver 1

Inspecting the Final Spacecraft State in Figure 34, we see that it compares well with the desired final spacecraft state in Table 8. We can move on with the final rendezvous maneuver.

Targeting the Rendezvous

Per Table 8, a final burn of 214.36 m/s will place our spacecraft into an orbit right alongside the Plock Orbiter 1. Because the results of our final burn were successful, we can simply go ahead and apply the final maneuver directly.

Our maneuver is listed in the “Final Orbit Injection Burn” segment of Table 8. We have the Universal Time, or “UT”, of this maneuver. To demonstrate the capability, let’s insert a Coast to that UT now.

Add a Coast as Event 13 in the usual way. When the dialog box opens, name the coast “Coast to Rendezvous” and set the color to Blue. Change the coast type to “Go to UT,” as this is the information we have available. Enter “641756991.25637” seconds as the Universal Time to go to and uncheck the optimization checkbox. It is usually unwise to optimize on UT directly as it can cause the optimizer to do funny things with time.

When you are done, save and close the dialog box. Your Mission Architect window should look similar to that shown in Figure 35.

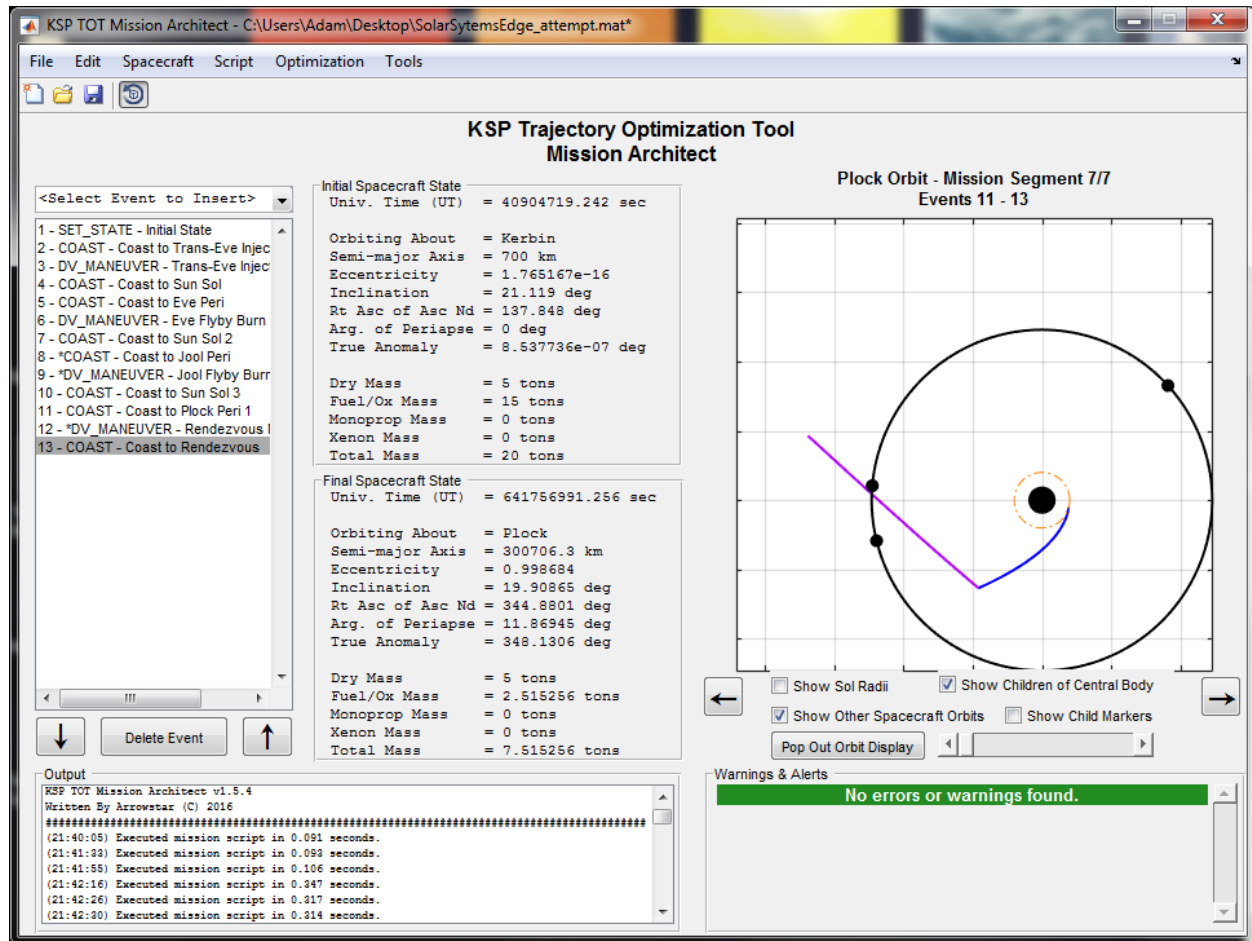


Figure 35: Mission Architect Window after Coast to Rendezvous

Finally, let's add the last burn. Insert a DV Maneuver as Event 14 into the script. Name it "Rendezvous Maneuver 2" and enter the prograde, normal, and radial components of the burn as shown in your RMS results. You may save and close the dialog when you've entered this information. Your DV Maneuver dialog box should appear as shown in Figure 36.

Edit DV Maneuver

Maneuver Name
Rendezvous Maneuver 2

Proscribed Delta-V (Orbital Vector)
LV-1R "Spider"

Delta-V Vector

Prograde	-156.715420000	m/s	<input checked="" type="checkbox"/>	Opt?	-1000.0	1000.0
Normal	-40.149650000	m/s	<input checked="" type="checkbox"/>	Opt?	-1000.0	1000.0
Radial	2.541150000	m/s	<input checked="" type="checkbox"/>	Opt?	-1000.0	1000.0

Save & Close Cancel

Figure 36: Rendezvous Maneuver 2

Consider the final state of the spacecraft now, as shown in Figure 37. You'll notice that the SMA, eccentricity, and inclination are all spot on with the Plock Orbiter 1 orbit. The other parameters are of less concern because we are in a circular, equatorial orbit. This suggests that we successfully met up with the Plock Orbiter 1. We will verify this fact in a moment.

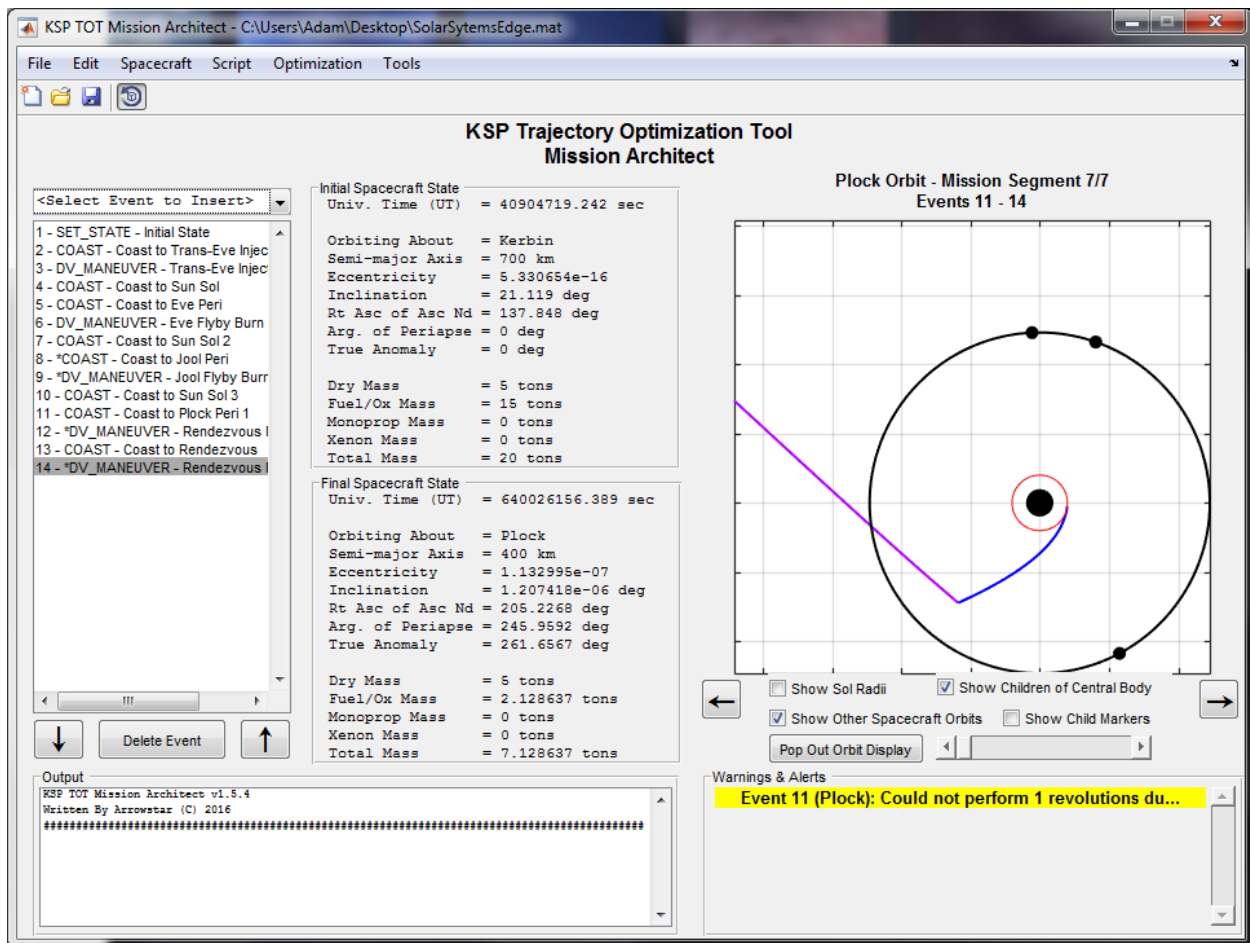


Figure 37: Mission Architect after Adding Rendezvous Maneuver 2

To conclude the rendezvous sequence, add a coast as Event 15 that goes to periapsis with a 1 additional revolution prior to going to periapsis. This simply gives a “pretty picture” and concludes the mission plan with a Coast event, which is typically cleaner.

In the next section, we’ll do some analysis to verify that we successfully rendezvoused with the Plock Orbiter 1 and look at our communications situation. However, congratulations are in order! Nice work on completing the “hard part” of this tutorial!

Lessons Learned

A variety of important skills, tips, and tricks have been learned in this section. To summarize, they are:

1. The Rendezvous Maneuver Sequencer can be used to generate highly accurate burn plans for rendezvousing with another spacecraft within the same sphere of influence.

Tutorial Part 7 – Analysis

In this section we will cover two common analysis tasks associated with our mission: verifying our rendezvous and looking at our communications network.

Rendezvous Verification

A simple test for ensuring that the spacecraft has rendezvoused with its target is to use Graphical Analysis to look at the distance between the two spacecraft. Open Graphical Analysis using the Tools -> Graphical Analysis menu.

This will be a simple analysis. Set the start time to the time at which the first rendezvous maneuver occurs. This can be found by right-clicking on the event in the event list (here, Event 12) and selecting “View State After Selected Event.” Simply copy the UT from the dialog that appears into the Start Time text box in Graphical Analysis. In my case, that time is 640019708 seconds.

Now, ensure that the Plock Orbiter 1 is the listed Reference Spacecraft and change it to that if not. Finally, select the Dependent Variable “Distance to Ref. Spacecraft” from the list. Then select “Generate Plots” and wait. The analysis may take some time to compute because the “distance to reference spacecraft” dependent variable is somewhat computationally intensive.

When the analysis completes, a chart will appear. You can use the zoom buttons near the top to zoom in on the end of the plot to see that the distance line trends to zero. You can also use the datatip feature to select the last point in the chart and see the distance (the “Y” value). Your chart should look something like Figure 38, below.

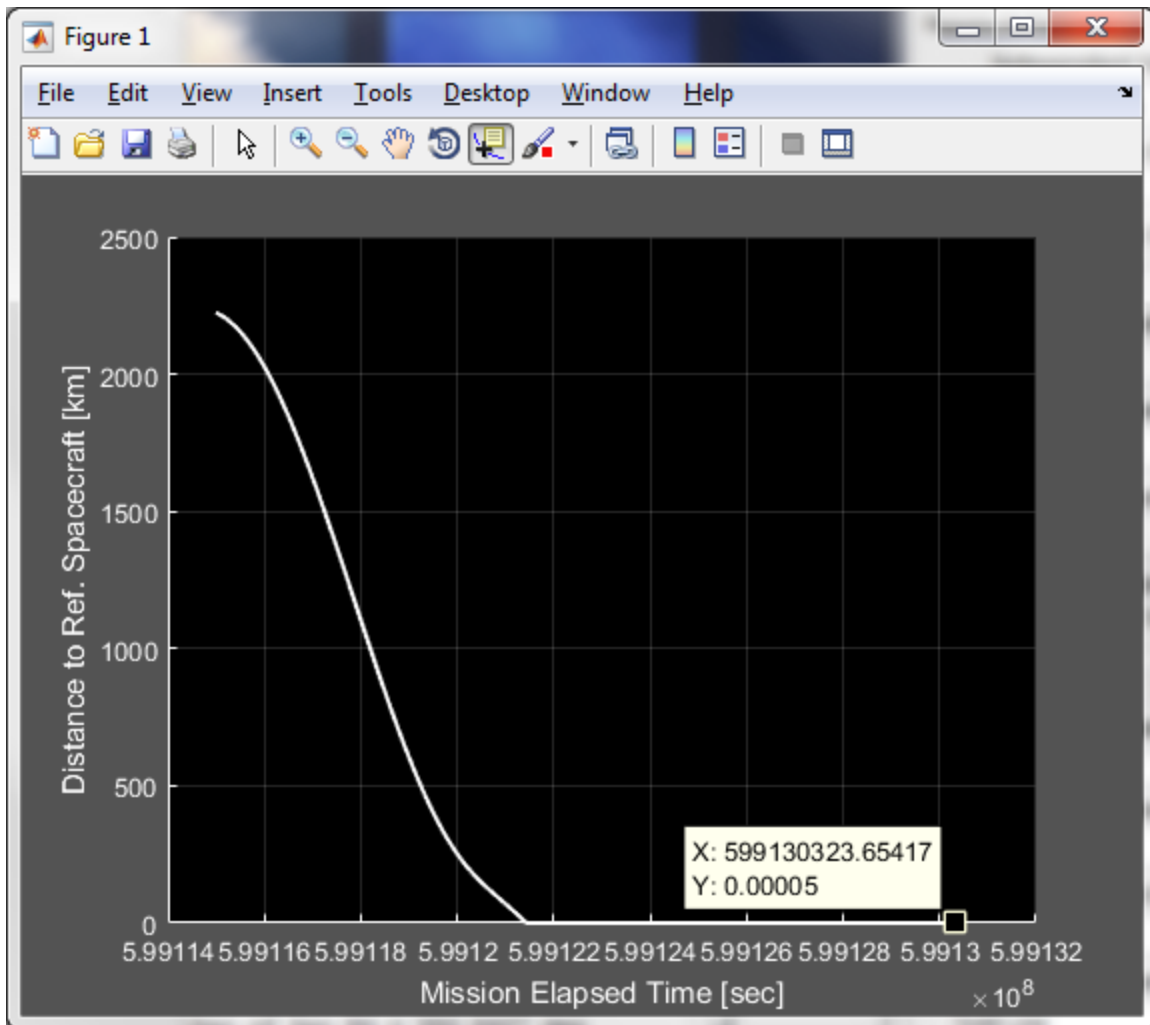


Figure 38: Graphical Analysis Rendezvous Verification

Given the miniscule distance shown in the plot, it is likely that our spacecraft is right on top of the Plock Orbiter 1. Of course, in practice the pilot of our spacecraft would begin maneuvering to dock or the like before this actually occurred, but those activities are outside of the scope of this tutorial.

Communications Network Analysis

The communications network analysis code in Mission Architect is very useful for determining when line of sight and spacecraft distance allow or deny communications access from one point to another. The tool is accessed from Tools -> Comm. Network Analysis. Open it now.

You may leave the network definition alone for now. It merely specifies which spacecraft are potential members of the communications network. By default, all ground stations, spacecraft, and the active vessel (the spacecraft we have been developing a mission plan for) are part of the comm network. If you have a large comm network and/ or many active vessels, deselecting satellites that wouldn't be used to send data or be capable of relaying to your terminus will speed up the analysis run.

Set the origin of the comm network to “KSC” and the terminus (end point) to the active vessel. Leave the other values as default. Push the “Perform Network Analysis” button, cancel the file selection dialog box that appears (it is used for saving results to file), and wait for the analysis to complete. It will likely take some time as performing the network analysis is a fairly CPU intensive task as well.

When the simulation completes, you will be presented with a chart similar to that shown below in Figure 39.

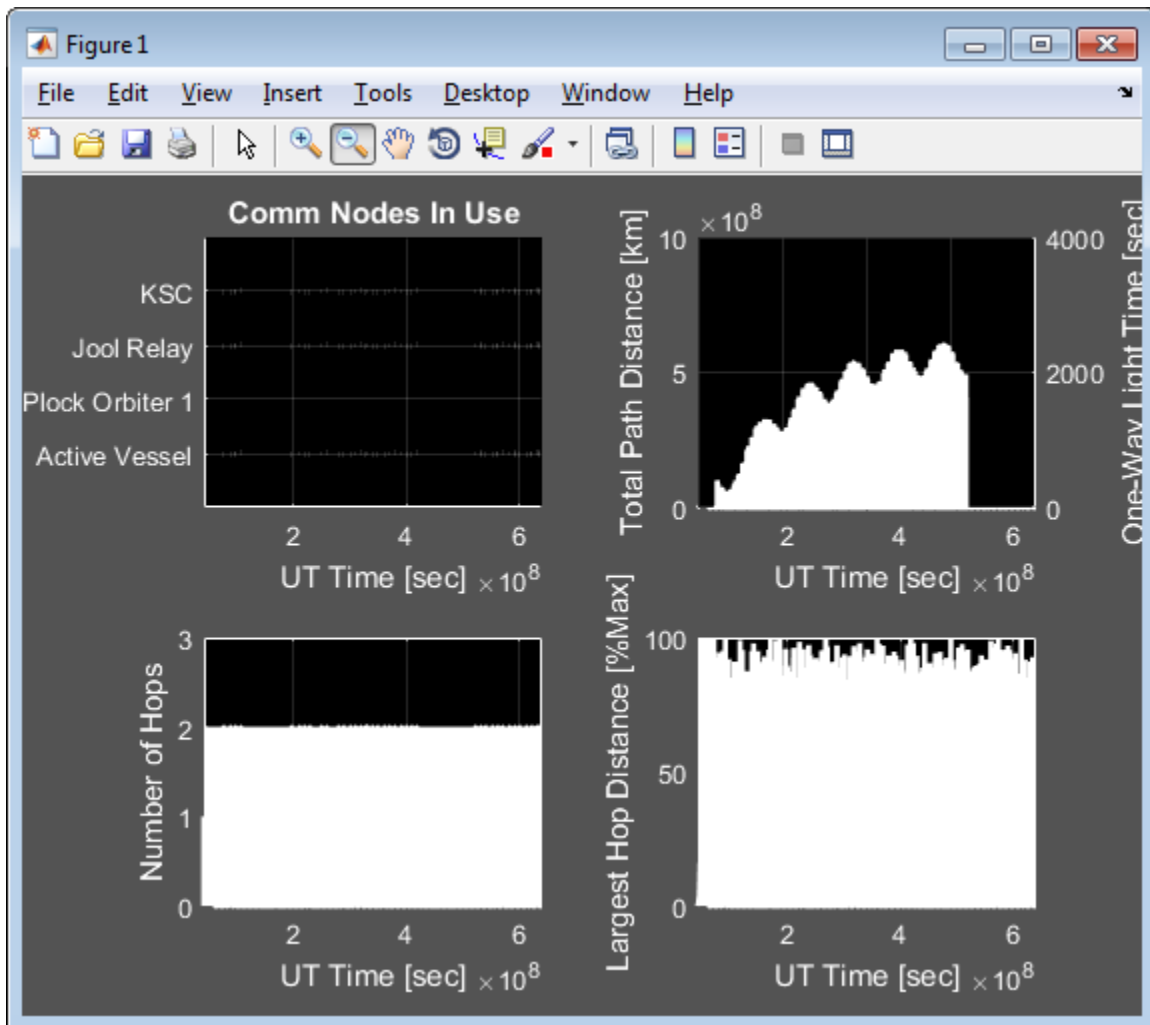


Figure 39: Comm Network Analysis Results

There are four plots on this graph. The upper left plot shows which communication nodes are in use at any given time. The plot here is a bit faint but more information can be gleaned by enlarging the image to see the points which represent in-use nodes.

The upper right plot shows the total distance the signal must travel and light time delay caused by that traversal. Note that there is a rendering glitch within MATLAB itself that occasionally causes plot points to not render correctly in the presence of a plot with many points, as shown here and in the upper left plot.

The lower left plot shows the number of hops required to transmit the signal from the origin to the terminus. This value is always one less than the total number of nodes in use at any given time.

The lower right plot shows the distance of the largest hop in terms of a percentage of that hop's maximum range. This lets you know when your communications are being hampered by distance and not line of sight.

And that's all there is to that! The communications network analysis function is a simple but very useful tool for determining communications network availability, particularly with KSP mods such as RemoteTech.

Appendix A – Glossary of Terms

Term	Definition
MFMS	“Multi-Flyby Maneuver Sequencer,” a tool in KSPTOT for finding and executing flyby opportunities.
RMS	“Rendezvous Maneuver Sequencer,” a tool in KSPTOT for determining rendezvous burn plans.
SoI / SOI	“Sphere of Influence,” the imaginary spherical volume around a celestial body in which that body’s gravity is dominant.

Appendix B – MFMS Results

<div>Hyperbolic Departure Orbit from Kerbin</div> <div><div>Semi-major Axis = -2004.600 km</div><div>Eccentricity = 1.3991</div><div>Inclination = 21.119 deg</div><div>Right Ascension of AN = 137.848 deg</div><div>Argument of Periapse = 360.000 deg</div></div> <div>Inbound Hyperbolic Flyby Orbit to Eve</div> <div><div>Semi-major Axis = -1224.660 km</div><div>Eccentricity = 1.6451</div><div>Inclination = 165.018 deg</div><div>Right Ascension of AN = 122.101 deg</div><div>Argument of Periapse = 159.994 deg</div><div>Periapse Radius = 790.000 km</div></div> <div>Outbound Hyperbolic Flyby Orbit from Eve</div> <div><div>Semi-major Axis = -561.249 km</div><div>Eccentricity = 2.4076</div><div>Inclination = 165.018 deg</div><div>Right Ascension of AN = 122.101 deg</div><div>Argument of Periapse = 159.994 deg</div><div>Periapse Radius = 790.000 km</div></div> <div>Inbound Hyperbolic Flyby Orbit to Jool</div> <div><div>Semi-major Axis = -67642.588 km</div><div>Eccentricity = 1.0917</div><div>Inclination = 5.413 deg</div><div>Right Ascension of AN = 117.899 deg</div><div>Argument of Periapse = 54.137 deg</div><div>Periapse Radius = 6200.001 km</div></div> <div>Outbound Hyperbolic Flyby Orbit from Jool</div> <div><div>Semi-major Axis = -65092.662 km</div><div>Eccentricity = 1.0952</div><div>Inclination = 5.413 deg</div><div>Right Ascension of AN = 117.899 deg</div><div>Argument of Periapse = 54.137 deg</div><div>Periapse Radius = 6200.001 km</div></div> <div>Inbound Hyperbolic Orbit to Plock</div> <div><div>Hyperbolic Excess Vel. = 0.783 km/s</div></div>	<div>Phase 1 Transfer Orbit (Kerbin -> Eve)</div> <div><div>Semi-major Axis = 11063958.385 km</div><div>Eccentricity = 0.23856</div><div>Inclination = 2.357 deg</div><div>Right Ascension of AN = 339.912 deg</div><div>Argument of Periapse = 192.654 deg</div><div>Period = 6753364.8284 sec</div><div>Departure True Anomaly = 167.346 deg</div><div>Arrival True Anomaly = 285.219 deg</div></div> <div>Phase 2 Transfer Orbit (Eve -> Jool)</div> <div><div>Semi-major Axis = 40292241.953 km</div><div>Eccentricity = 0.75812</div><div>Inclination = 4.094 deg</div><div>Right Ascension of AN = 308.351 deg</div><div>Argument of Periapse = 160.082 deg</div><div>Period = 46933891.078 sec</div><div>Departure True Anomaly = 349.308 deg</div><div>Arrival True Anomaly = 183.842 deg</div></div> <div>Phase 3 Transfer Orbit (Jool -> Plock)</div> <div><div>Semi-major Axis = 329683729.477 km</div><div>Eccentricity = 0.80511</div><div>Inclination = 1.655 deg</div><div>Right Ascension of AN = 69.125 deg</div><div>Argument of Periapse = 186.897 deg</div><div>Period = 1098501572.4198 sec</div><div>Departure True Anomaly = 36.302 deg</div><div>Arrival True Anomaly = 181.687 deg</div></div> <div><div>Kerbin Departure Date = Year 2, Day 109 10:25:19.242 (40904719.242 sec UT)</div><div>Eve Arrival Date = Year 2, Day 142 00:27:14.948 (43720034.948 sec UT)</div><div>Jool Arrival Date = Year 3, Day 77 10:19:45.129 (69675585.129 sec UT)</div><div>Plock Arrival Date = Year 21, Day 104 04:21:06.659 (639634866.659 sec UT)</div></div>
--	---

Burn Information to Depart Kerbin

Total Delta-V = 1.500 km/s
Prograde Delta-V = 934.697 m/s
Orbit Normal Delta-V = 1172.544 m/s
Radial Delta-V = 0.000 m/s

Burn True Anomaly = 137.848 deg

Burn Information to Depart Eve

Total Delta-V = 0.706 km/s
Prograde Delta-V = 706.254 m/s
Orbit Normal Delta-V = -0.000 m/s
Radial Delta-V = -0.000 m/s

Burn True Anomaly = 0.000 deg

Burn Information to Depart Jool

Total Delta-V = 0.008 km/s
Prograde Delta-V = 8.376 m/s
Orbit Normal Delta-V = 0.000 m/s
Radial Delta-V = -0.001 m/s

Burn True Anomaly = 0.000 deg
