

Kerbin to Laythe: A KSP TOT Mission Architect Tutorial

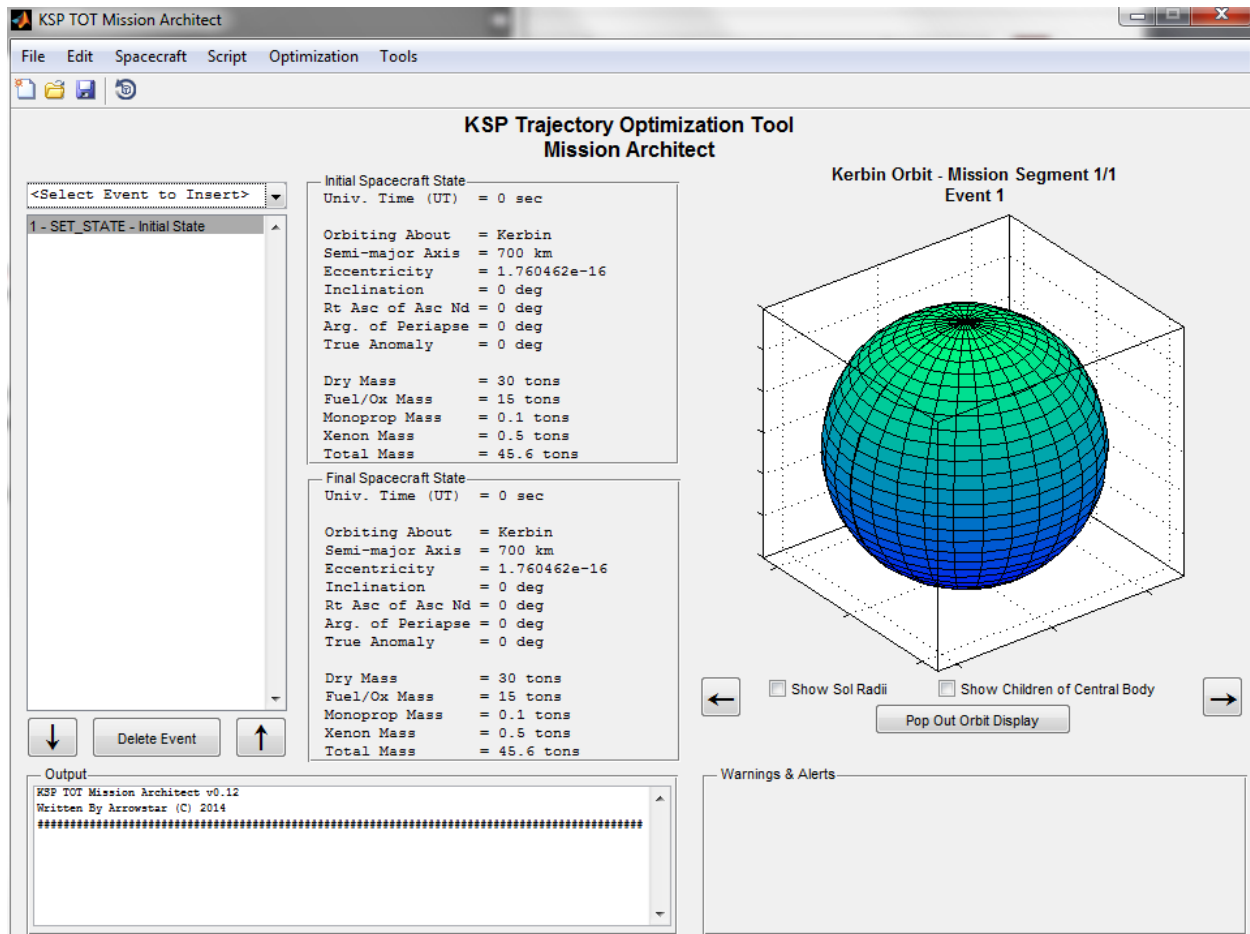
The purpose of this tutorial is to teach you how to plan a simple mission from Kerbin to Laythe. The final Laythe orbit will be circular and as equatorial as we can arrange for it to be in one burn. After completing the tutorial, you should have some idea about how to set up a mission plan, use the optimizer to goal seek, and constrain the mission plan.

This tutorial assumes familiarity with the main KSP TOT functionality (the porkchop plot).

Step 1: Opening Mission Architect

Start the KSP Trajectory Optimization Tool by double clicking on the KSPTOT.exe executable. After it has started, start Mission Architect by selecting it from the menu: **Tools -> Mission Planning -> Mission Architect**.

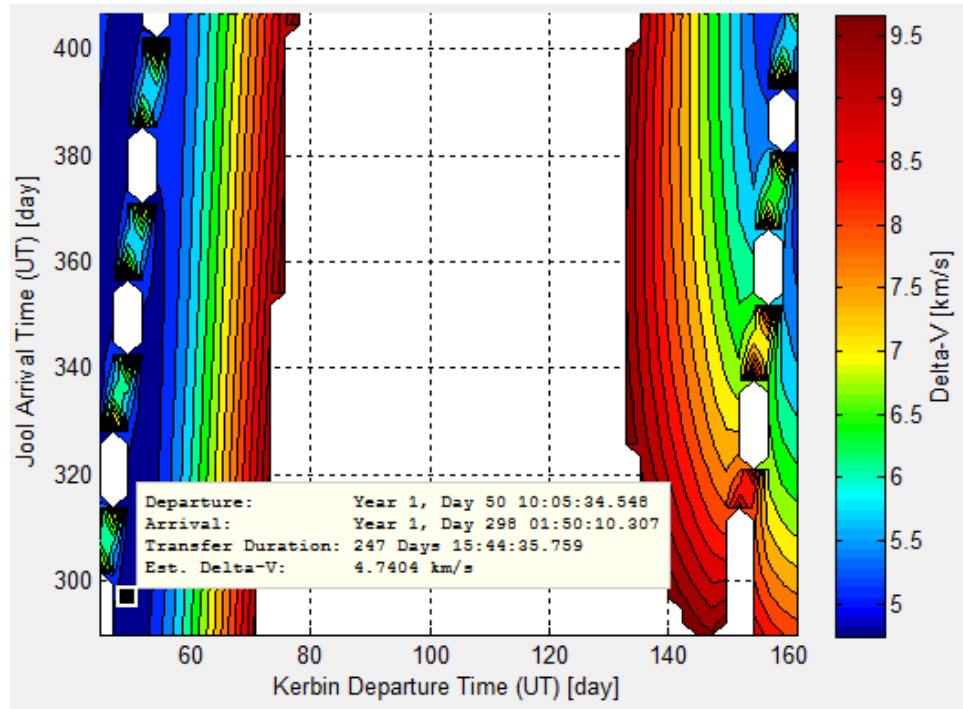
You should shortly be presented with an empty mission plan in the Mission Architect (“MA”) GUI.



Step 2: Use KSP TOT to determine basic trajectory parameters.

I know we just opened MA, but for the moment, minimize it. Go back to the main KSP TOT window and use the porkchop plot application to find a trajectory from Kerbin to Jool (the parent of Laythe).

One potential solution appears as such:



Using the Compute Departure function with a 700 km equatorial orbit, the transfer orbit and burn look like this:

Transfer Orbit	Burn
Transfer Orbit about Sun	Burn Information to Depart Kerbin
Semi-major Axis = 41731373.672 km	Total Delta-V = 2.39589 km/s
Eccentricity = 0.676	Prograde Delta-V = 1794.09900 m/s
Inclination = 4.822 deg	Radial Delta-V = 0.00005 m/s
Right Ascension of AN = 347.755 deg	Orbit Normal Delta-V = 1587.92407 m/s
Argument of Periapse = 350.269 deg	
Kerbin Depart True Anomaly = 9.731 deg	Burn True Anomaly = 307.69239 deg
Jool Arrive True Anomaly = 174.285 deg	Burn Time Past Peri. = 1673.61451 sec
	Burn Time Before Peri. = 284.51393 sec
Departure Date = Year 1, Day 50 15:57:41.010 (4291061.010 sec UT)	Initial Orbit Period = 1958.12844 sec
Arrival Date = Year 1, Day 300 07:55:56.800 (25862156.800 sec UT)	
Duration = 249 Days 15:58:15.790	

Hang on to these parameters, we will need them momentarily when we get back to Mission Architect.

Step 3: Setting up the MA Initial State

Restore the Mission Architect window and minimize the other KSP TOT windows. Don't close the other windows, we may need them in the future.

In the mission script box on the left, select the SET_STATE event and double-click it.



Figure 1: The mission script

A box will appear that allows you to edit the initial state of the vehicle.

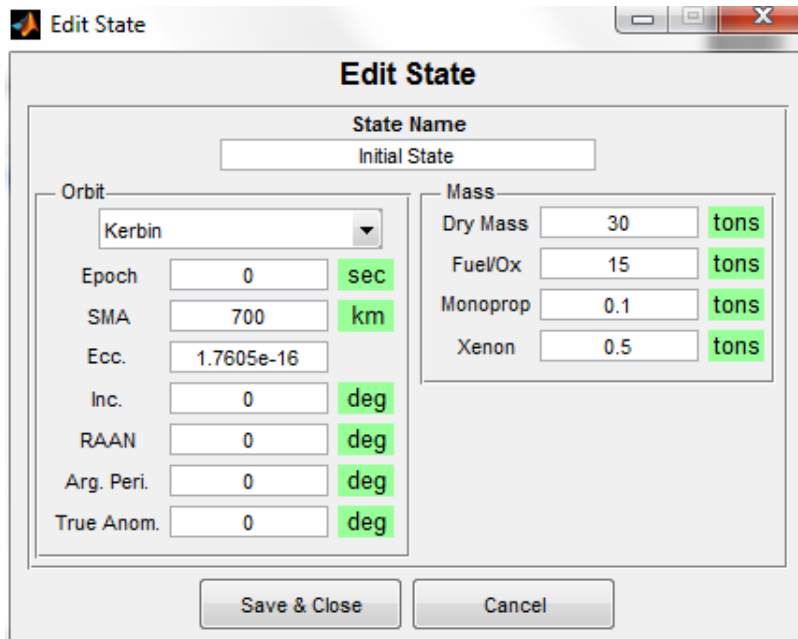


Figure 2: The Initial State dialog box

Leave the orbit as is with one exception: set the true anomaly to a few degrees before our burn true anomaly of 307.69 degrees. Use, for example, 300 degrees even. In addition, set the epoch to a few minutes prior to the Departure Date of 4291061.010. Use, for example, 4290061.010. Your Edit State box should appear thusly after you make these changes:

State Name	
Initial State	
Orbit	
Orbit	Kerbin
Epoch	4290061.01 sec
SMA	700 km
Ecc.	1.7605e-16
Inc.	0 deg
RAAN	0 deg
Arg. Peri.	0 deg
True Anom.	300 deg
Mass	
Dry Mass	30 tons
Fuel/Ox	15 tons
Monoprop	0.1 tons
Xenon	0.5 tons

Save & Close Cancel

Figure 3: Populated Initial State

Tap “Save and Close.” Notice how the Initial Spacecraft State fields on the main MA GUI update to reflect the new initial state. Mouse over the Initial Spacecraft State fields to learn more about the state.

Step 4: Create Coast to Burn Position

We need to insert a coast to the burn position in our orbit. Coasts are events in which the spacecraft travels through space according to two-body motion with the patched conics model of Kerbal Space Program. No engines fire during a coast. We know from the pork chop plot analysis we did that the burn will be at roughly 307.69239 degrees of true anomaly. To insert a coast, select the event in the mission script list you wish to insert the coast **AFTER** (in this case the only event we have, the Initial State event) and select **Coast** in the dropdown box above the mission script.

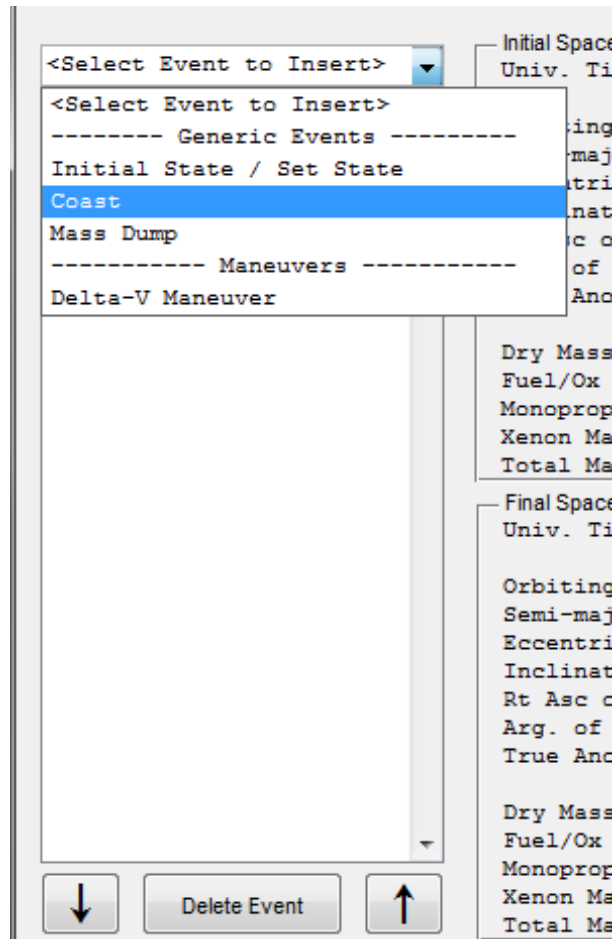


Figure 4: Select the Coast Event to Insert

A dialog box will appear.

Figure 5: The Insert Coast dialog box

Name the Coast “To TJI Burn”. TJI here stands for “Trans-Jool Injection,” which is effectively what our upcoming burn is. Enter the burn true anomaly of 307.69239 degrees in the True Anomaly box. Leave the “Revs Prior to Coast” at 0. This field allows you to go around the body N times prior to coasting to the target true anomaly if your orbit is not hyperbolic (if your orbit is hyperbolic, the field is ignored).

Note that other Coast types exist. Some types allow you to specify a value, such as coasting to a particular UT or delta time. Others only go to one place, such as “go to periapsis.”

The “Opt?” check box below the True Anomaly tells the optimizer to optimize the value of this coast. The two fields below the checkbox are the bounds on the optimized true anomaly. The left box is the lower bound and the right box is the upper bound. It is **always** in your best interest to make these bounds as tight as you can without excluding the solution. Bounds which are too large will yield poor solutions. Here, since we know the true anomaly should be nearly 307 degrees, we can enter 300 in the left box as a lower bound and 320 in the right box as an upper box. This should give the optimizer some room to wiggle without having too much freedom.

Finally, leave the Reference Body dropdown empty. Tap “Save and Close.”

Your completed Coast should look like this:

Create Coast

Coast Name
To TJI Burn

Go to True Anomaly

True Anomaly
307.69239 deg

Revs Prior to Coast
0

☒ Opt?

300 320

Reference Body

Save & Close Cancel

Figure 6: Completed TJI Coast

You should notice a little red arc appear in your trajectory visualization on the right of the MA GUI.

Step 5: Create TJI Burn

In the mission script list, select the COAST we just created and then insert a “Delta-V Maneuver” in the same manner we inserted the coast, using the drop down box. You should be presented with a screen like this:

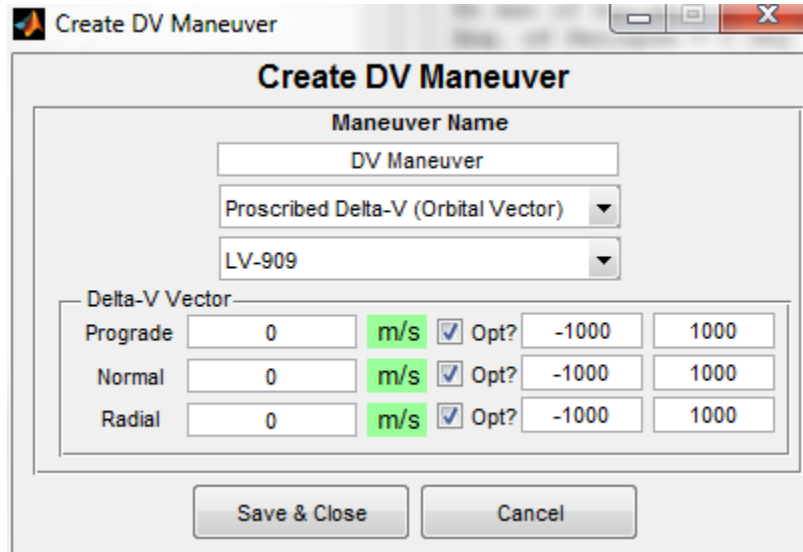


Figure 7: Create DV Maneuver Dialog

First, name the maneuver “TJI Burn” in the upper text box. Leave the next drop down as “proscribed delta-v (orbital vector)”. This means we’re dictating the delta-v vector and we’re specifying that vector in Prograde/Normal/Radial components. We could also use X/Y/Z components, but those aren’t as nice.

Use the porkchop plot information we obtained earlier to populate the delta-v vector components. Be careful that you enter the components in the correct order, as the order of the normal and radial components is different in different parts of KSP TOT. (Aside: MechJeb uses one convention and another maneuver node editor uses another convention. What’s a programmer like me to do? 😊)

Now, let’s set the bounds. We know the actual solution should be close to the one we computed elsewhere. On the prograde component, let’s use 1600 m/s as the lower bound and 2000 m/s as the upper bound. Likewise, on the normal component, use 1400 m/s and 1800 m/s. On the radial component, let’s use -10 m/s and 10 m/s. We don’t want a large radial component, so we restrict the solution thusly.

Make sure those “Opt?” check boxes are still checked, by the way!

Your completed burn should look like this:

Create DV Maneuver

Maneuver Name

TJI Burn

Proscribed Delta-V (Orbital Vector)

LV-909

Delta-V Vector

Prograde	1794.099	m/s	<input checked="" type="checkbox"/>	Opt?	1600	2000
Normal	1587.92407	m/s	<input checked="" type="checkbox"/>	Opt?	1400	1800
Radial	5e-05	m/s	<input checked="" type="checkbox"/>	Opt?	-10	10

Save & Close Cancel

Figure 8: Completed TJI Burn Dialog

Tap “Save and Close.”

By the way, the engine listed under the Burn Type (here, “LV-909”) only dictates the Isp of the engine used and thus the amount of fuel used in the burn. It does not dictate perform in any other way, as all burns are modeled as impulsive in Mission Architect. You can add, delete, and modify engines by going to the Spacecraft menu in the main Mission Architect GUI.

When you get back to the main MA GUI, you should notice that nothing has changed in the orbit visualization, but the Final Spacecraft State is now far, far different than the initial! This is good. It means the burn did something to our orbit.

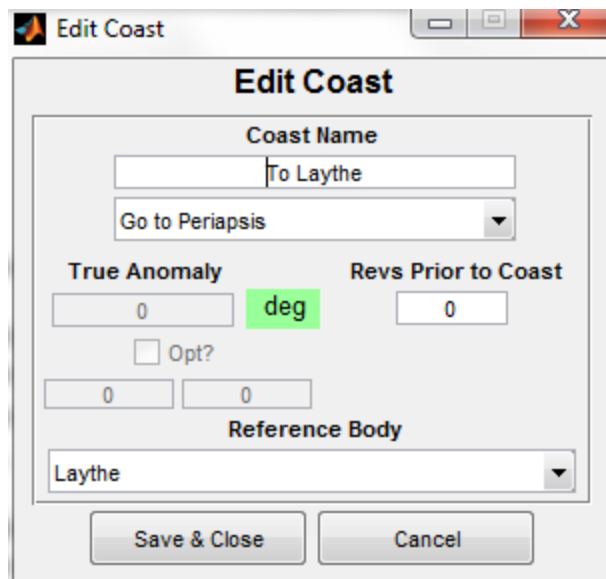
Step 6: Create Coast to Laythe

Insert another Coast in the manner you did for Step 4. (You're a pro at inserting events now, right?)

This coast is special. Unlike the previous coast, we're going to try to coast directly to the periapse of our target, Laythe. Change the Coast Type from "Go to True Anomaly" to "Go to Periapsis." Notice that the optimization parameters gray out when you do so. It is not possible to optimize events that go to fixed points in the orbit. Leave "Revs Prior to Coast" as 0 again.

Here, however, we need to specify the body we want to go to the periapse of. Select Laythe in the Reference Body drop down. If you leave this blank, the script will go to the first periapse it encounters regardless of body.

Name the Coast "To Laythe." Tap "Save and Close." Your coast dialog should look like this:



Now we're cooking with gas! Notice that the orbit visualization on the right of the main MA GUI just did a bunch of stuff. You should now see a nice departure orbit from Kerbin. Use the arrow buttons under the visualization to see each segment of the mission.

If you scroll through each segment (effectively, each Sphere of Influence), you'll notice that while you hit Jool's Sol, you missed Laythe! The final orbit is, in fact, be around the Sun. What gives?

Since the burn we planned didn't hit Laythe, the propagation code realized it was never going to hit Laythe and stopped propagating once it got around the sun. This is a safety feature to prevent the code from propagating forever.

No worries, though. In the next step, we're going to use the optimizer to hit Laythe.

Step 7: Optimize the Mission Parameters

Now it's time to figure out how to get to Laythe. Under the Optimization menu, select "Optimize Mission." Alternatively, just hit Ctrl+O on your keyboard.

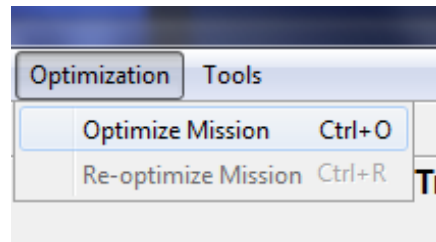


Figure 9: Optimize Mission

A dialog like the following should come up.

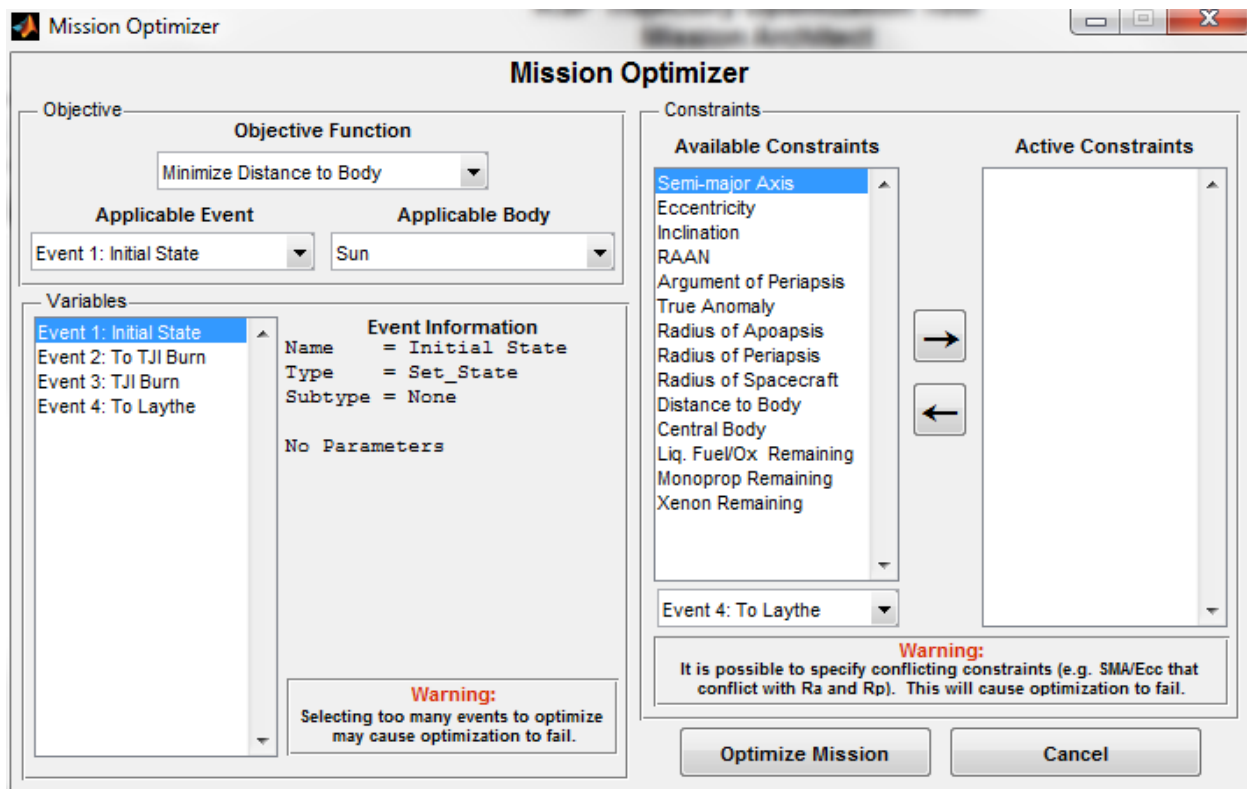


Figure 10: Optimizer Dialog Box

There's a lot here, and we'll get to most of it eventually. For now, though, let's talk about the Objective Function. The objective function is the code that tells the optimizer how to vary the solution to get what you want. So what do we want here? We want to go to Laythe. How do we do that? Well, we can minimize the distance to Laythe for starters.

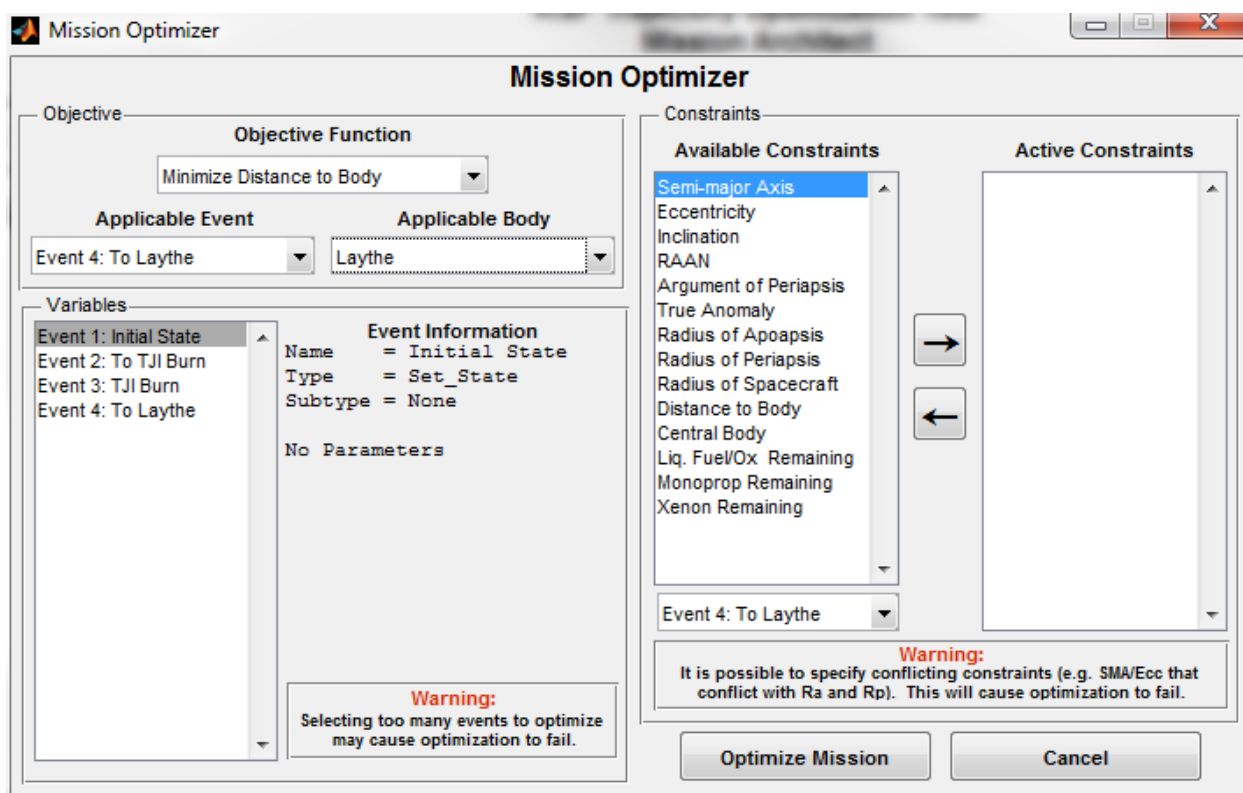
In the Objective box, change the Objective Function to Minimize Distance to Body. Under the Applicable Event, let's select our final coast, "To Laythe." The optimizer will attempt to minimize the distance the

closest point on this arc gets to Laythe. Note: If you leave this at Event 1 (Initial State), the optimizer **will not do anything!** It's very important that this gets update. Finally, change the Applicable Body to (you guessed it) Laythe.

The Variables box underneath the Objective box is just a viewer to show you what's being optimized and what the current values of the variables are. Select an event and check out the data on that event.

Let's ignore constraints for now. When you're first trying to target a body, don't input constraints. They just make it harder for the optimizer to hit the target body. You can always come back and optimize with constraints later, after you've got a good initial solution that actually gets to Laythe.

Your final optimization dialog box should look like this:



The major take away from this section here is this: **the optimizer does not just get used to make the mission more efficient. It can also be used to target bodies, minimize eccentricity and inclination, and otherwise goal seek. You make the mission do what you want by using the optimizer.**

Once you've read that bolded text a few times, tap "Optimize Mission" and say a prayer. The following box should come up shortly after you do so:

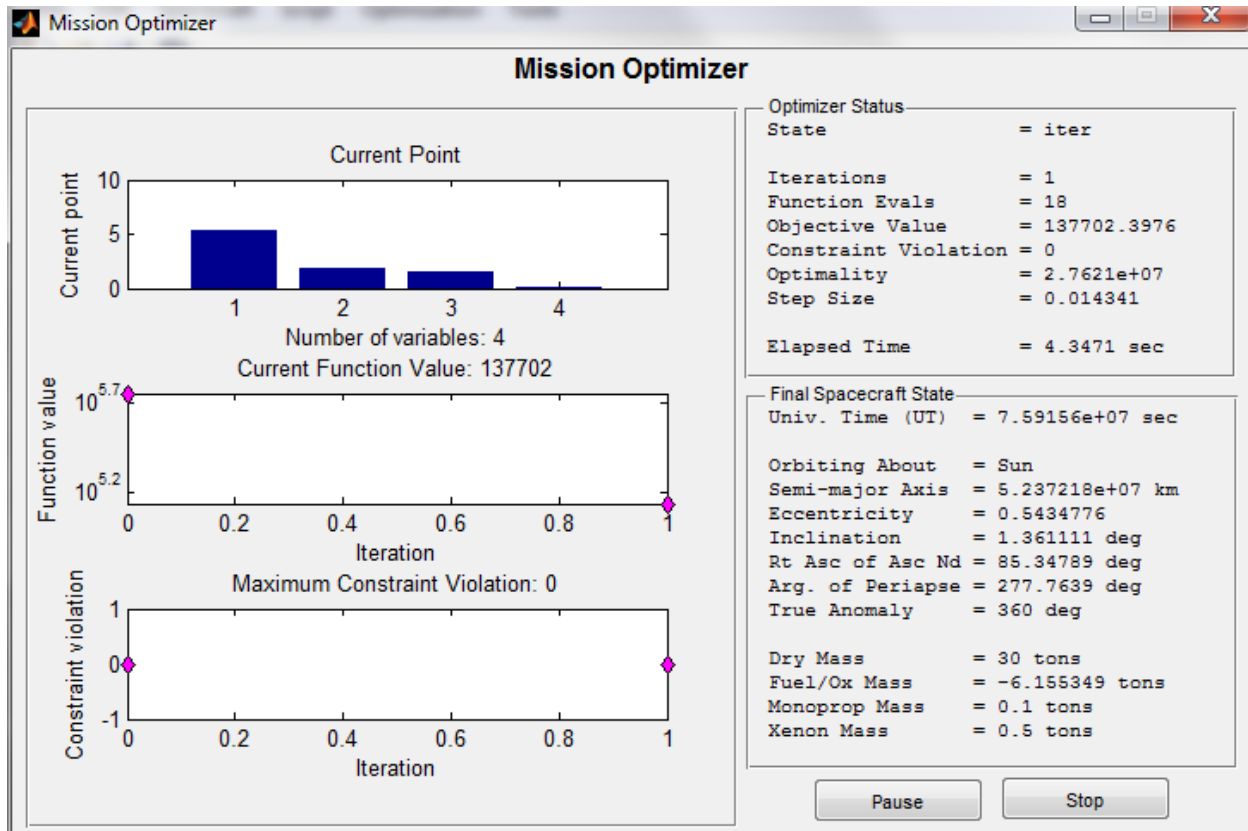


Figure 11: Mission Optimizer

Watch the optimizer do its thing. It may take a few minutes as the SOI transition search code can be a bit slow in the vicinity of Jool.

Here are some things to look for when the optimizer is running:

1. The Objective Function should be decreasing towards zero, though it may take time to get there.
2. The constraint violation is 0, as we don't have any constraints other than bounds on the variables right now.
3. The final spacecraft state should show the body being orbited as Laythe at some point.

Wait until either the optimizer finishes doing its thing or once the Current Function Value (the closest distance to Laythe) gets really small, say under 10 km, just hit the stop button. Your screen might look something like this:

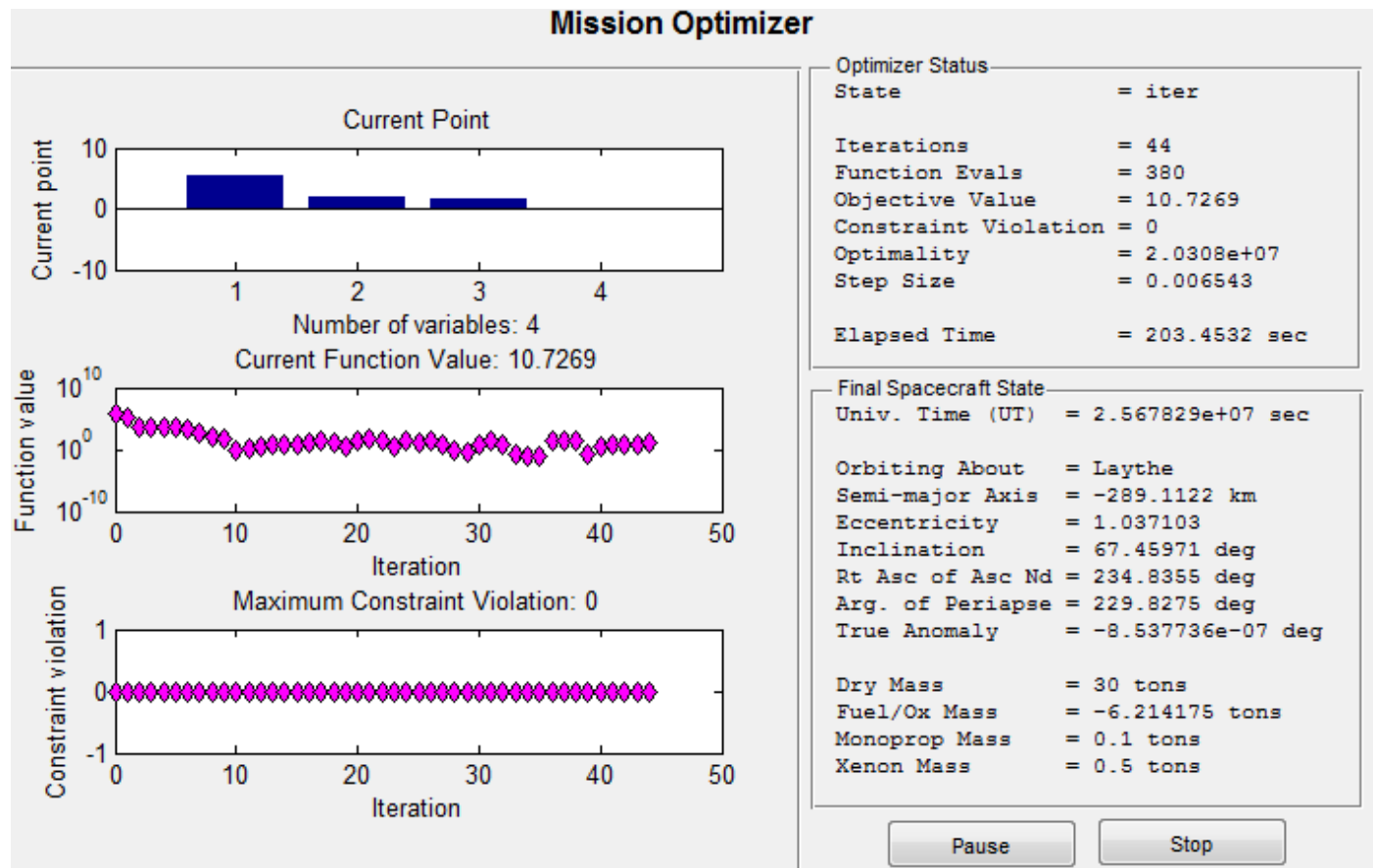
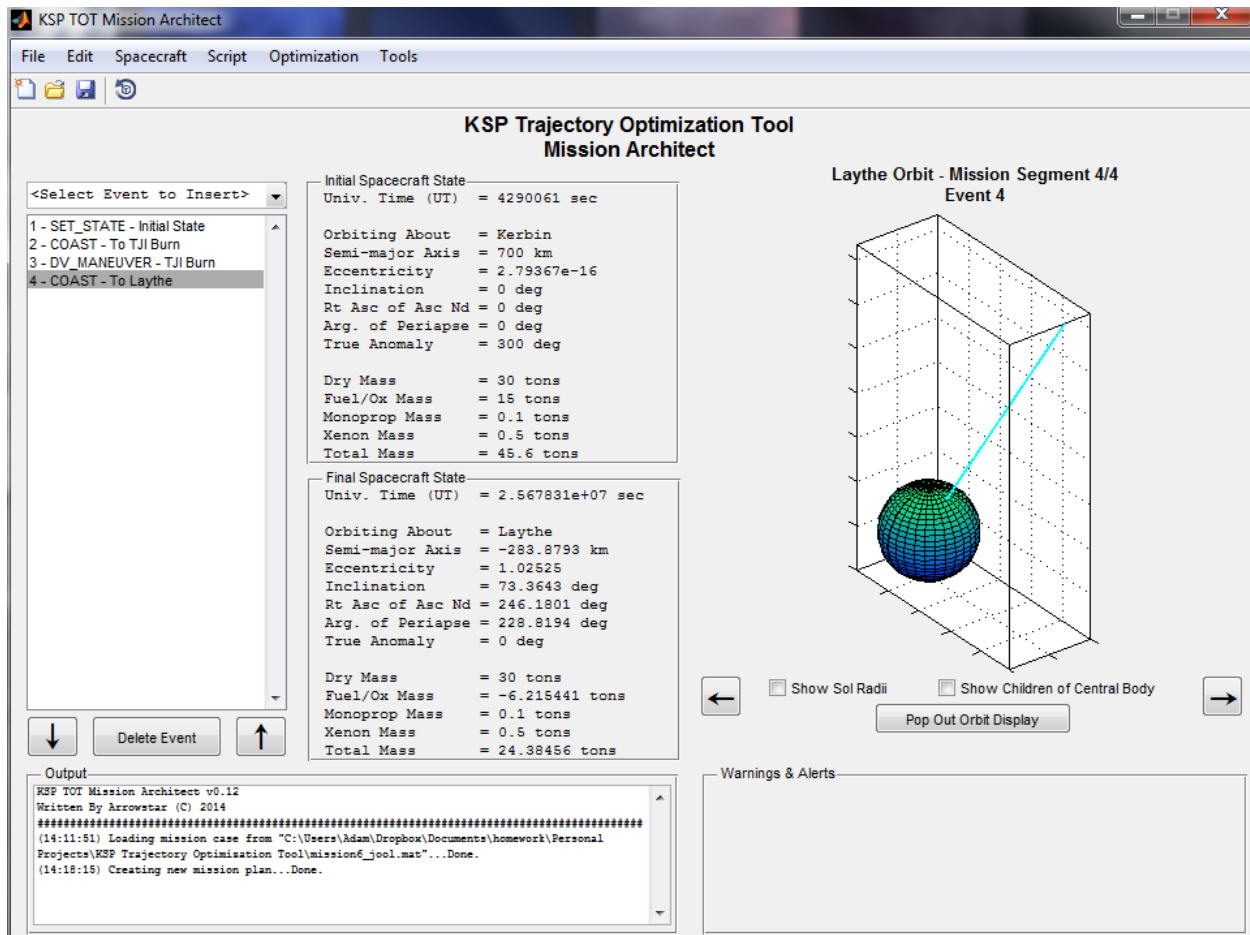


Figure 12: Final Mission Optimizer Screen

A box will appear asking if you want to keep the solution. Tap “Yes.”

In the orbit visualization, use the Right Arrow to view Mission Segment 4/4. You should have something like this:



Well, we said we wanted to get close to Laythe, and now we're going to smack right into it. But that's better than missing it completely, right? Right?

Step 8: Using the Optimizer to Minimize Inclination at Laythe

Open up the Mission Optimizer window again using either the menu or Ctrl+O. It's time to use constraints to stop the optimizer code from forcing Jeb, Bill, and Bob to commit rapid suicide into the Laythe oceans.

We'd like to add three constraints initially. The first is the Central Body constraint. This constrains the body of the select event to a particular celestial body in the KSP universe. The second is the inclination constraint, which we will use to minimize inclination. The third is the Radius of Periapse constraint, which we'll use to keep from slamming into the surface.

Note: Constraints are evaluated at the end of the event to which they are applied.

To add the central body constraint, select Central Body in the Available Constraints list. Make sure "Event 4: To Laythe" is selected in the drop down below. This is the event that the constraint will be applied to. Tap the Right Arrow. A box will appear:

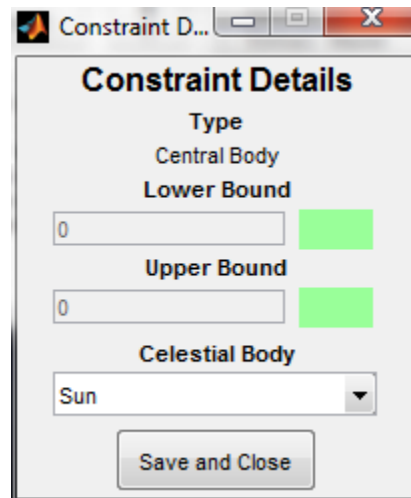
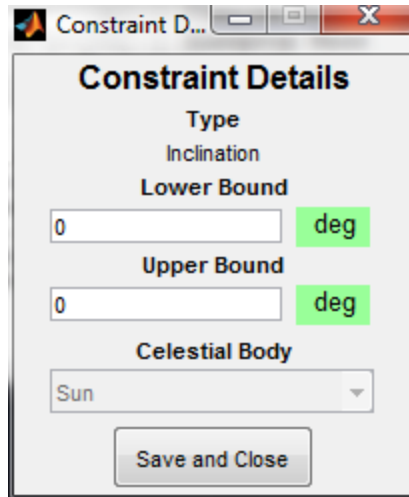


Figure 13: Central Body Constraint

Here, all we have to do is change "Sun" to "Laythe" in the Celestial Body dropdown. Do so and tap "Save and Close."

Notice how the constraint shows up under the Active Constraints list now. Let's do the same with the inclination constraint. Select Inclination under Available Constraints, make sure Event 4 is selected, and tap the Right Arrow.

This box is a bit different. It wants a lower and upper bound and you cannot specify a celestial body. Here, since we'd like the orbit inclination around Laythe to get as close to 0 as possible, specify 0 and 0 for the lower and upper bounds. Tap "Save and Close."

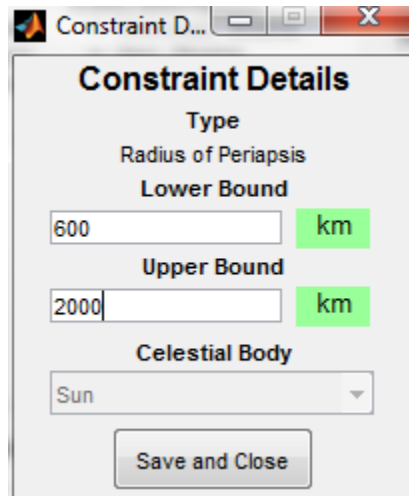


The screenshot shows a window titled "Constraint D..." with a close button. Inside, the "Constraint Details" dialog has the following fields:

- Type:** Inclination
- Lower Bound:** A text box containing "0" followed by a green button labeled "deg".
- Upper Bound:** A text box containing "0" followed by a green button labeled "deg".
- Celestial Body:** A dropdown menu with "Sun" selected.
- Save and Close:** A button at the bottom.

Figure 14: Inclination Constraint

Add the Radius of Periapse constraint. Use 600 km as the lower bound and 2000 km as the upper bound. **Never specify Inf or -Inf as constraint values, this will break the code.** Only use real, finite values here.



The screenshot shows a window titled "Constraint D..." with a close button. Inside, the "Constraint Details" dialog has the following fields:

- Type:** Radius of Periapsis
- Lower Bound:** A text box containing "600" followed by a green button labeled "km".
- Upper Bound:** A text box containing "2000" followed by a green button labeled "km".
- Celestial Body:** A dropdown menu with "Sun" selected.
- Save and Close:** A button at the bottom.

Figure 15: Radius of Periapse Constraint

This should be all set for now. Your constraints should look like this:

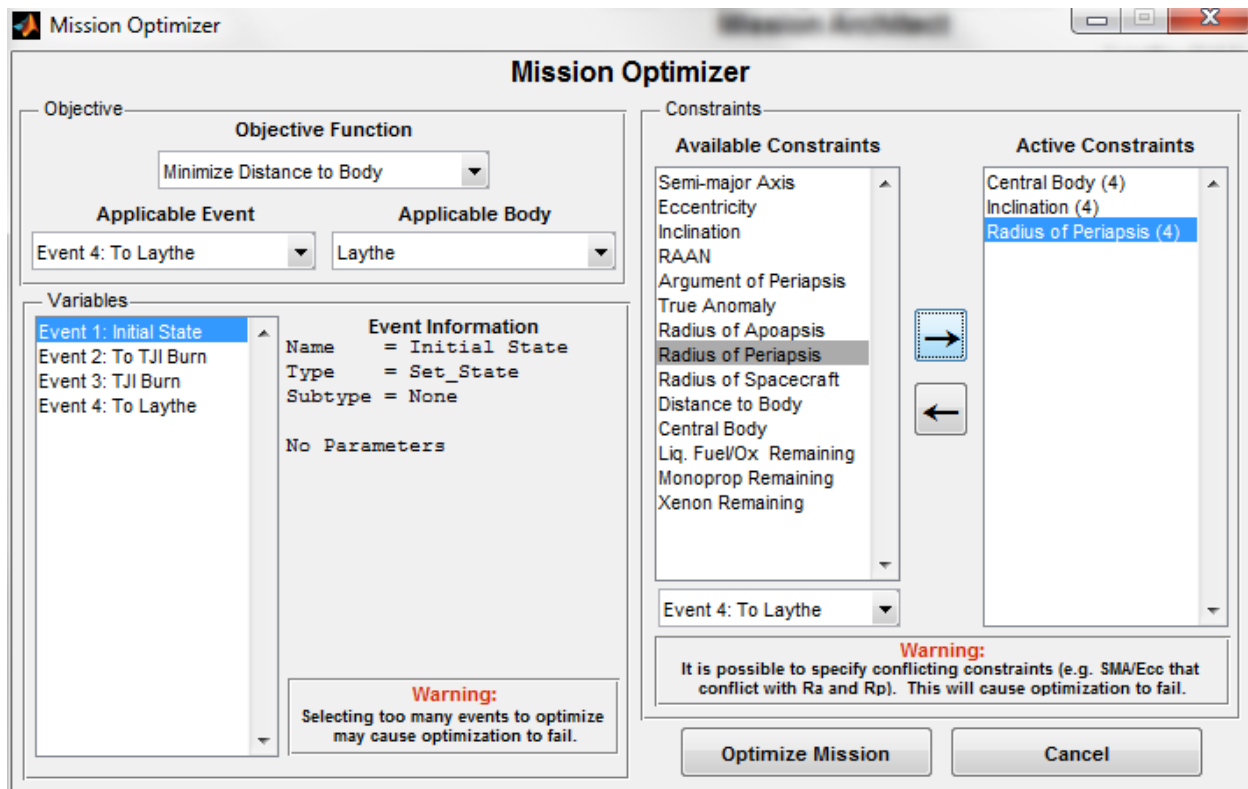


Figure 16: Optimize Mission GUI: Round 2

Notice how the constraints all have a parenthetical number next to them: this is the applicable event number and it should be “4” in all cases here.

Make sure we’re still Minimizing Distance to Body (objective function). Tap “Optimize Mission” and start praying again. Note that because we have constraints, the optimizer code is going to run a bit more slowly. It has to evaluate the constraints, which requires one more function evaluation and some additional math after that.

Just a note: we are going to insert a maneuver later that will help with minimizing inclination and all that. However, it’s best to minimize the amount of inclination we need to change up front so we don’t need to do everything while approaching Laythe periapse, which would be very inefficient. If we can even get the inclination down to 30-40-50 degrees, that will be a big help.

Feel free to stop the optimizer as soon as the objection function value is greater than 600 and the inclination is less than 60 degrees. Keep the solution. Again, the optimizer is used as a goal seek tool here, nothing more.

My orbit visualization at Laythe now looks like this. You should have something similar.

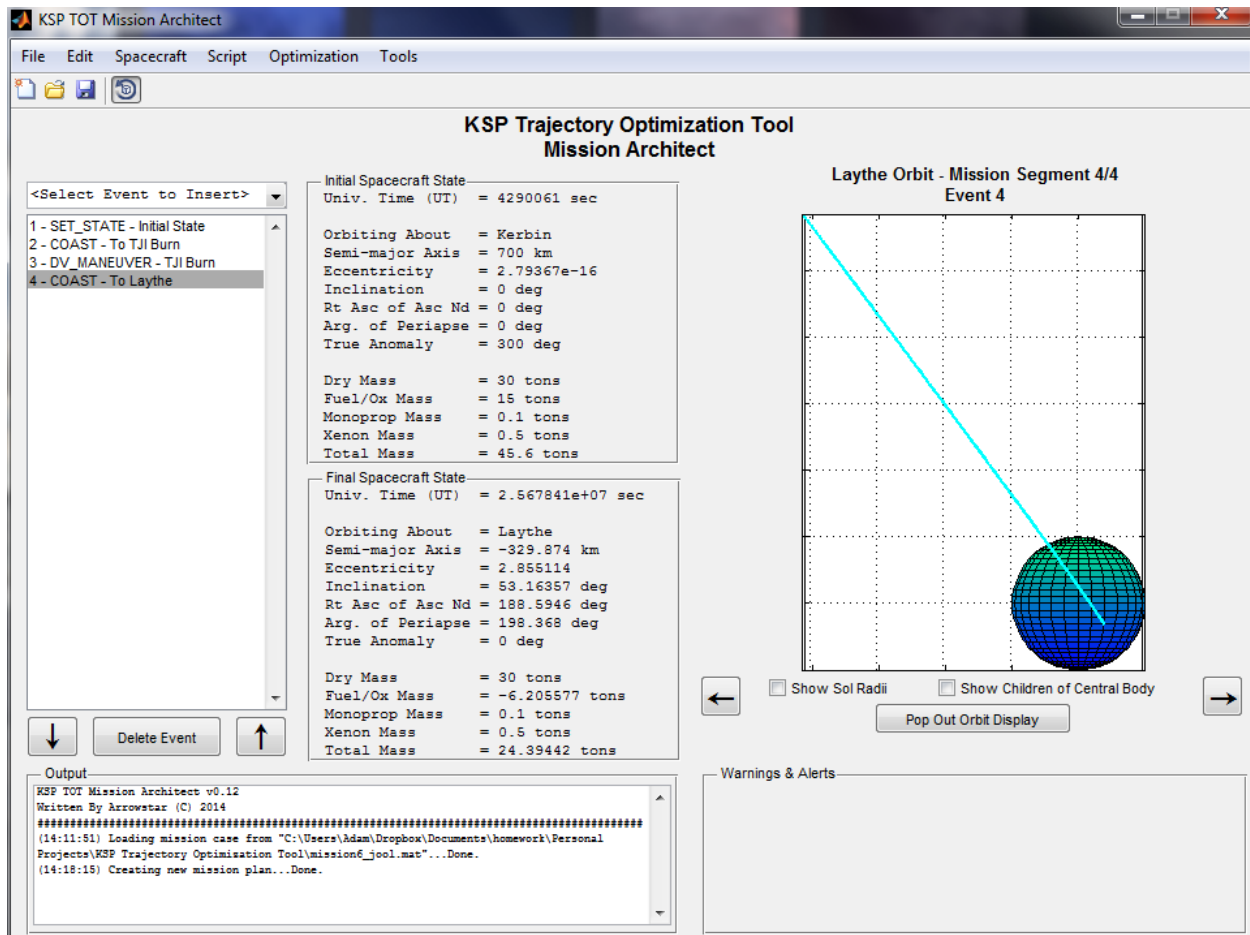


Figure 17: After Optimization Round 2

Step 9: Circularize Orbit Around Laythe

So at this point we've got an orbit around Laythe, but it's hyperbolic and pretty inclined. We'd like a nice, low, circular orbit with small inclination. We can use the optimizer to do this too!

First, realize that we probably aren't going to get much out of adjusting our Kerbin-based TJI burn anymore. It's too far away and there's too much coast for the optimizer to handle it well. So let's turn off those parameters so the optimizer doesn't see them.

Double click the "To TJI Burn" coast. Uncheck the "Opt?" checkbox. Tap "Save and Close."

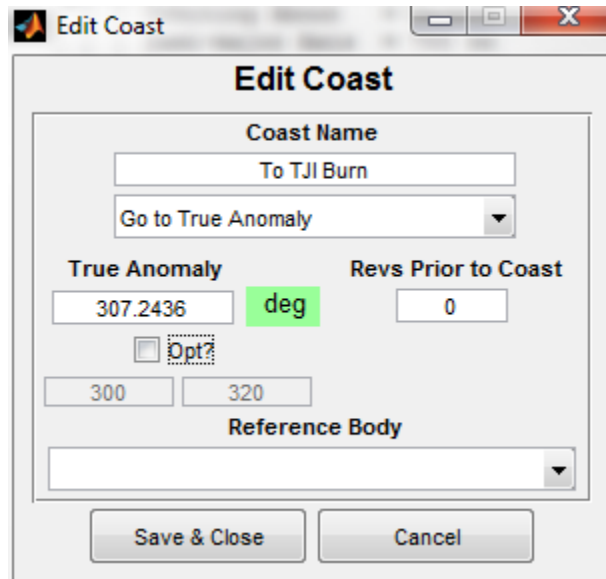


Figure 18: Turn off Coast Optimization

Double-click the "TJI Burn" event and do the same with the three "Opt?" boxes there:

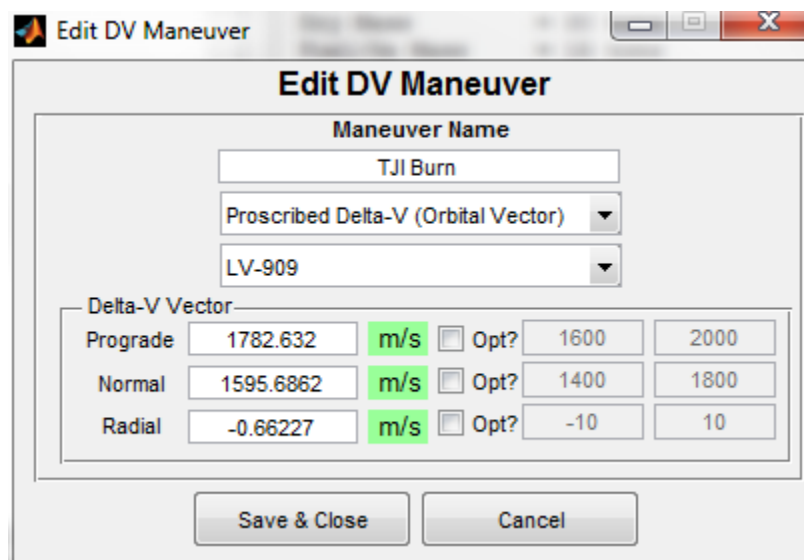


Figure 19: Turn Off Burn Optimization

The next step is to add more events that we *can* optimize. Let's add a burn right after the "To Laythe" Coast. Call it "Laythe Injection".

Leave the Delta-V vector at all 0 for now (except prograde, change that to -2000 m/s), since we don't actually know how to change the orbit to what we want in one burn (NOTE: you could use the two-burn orbit change code in KSP TOT to get a good approximation of these values if you wanted to here).

However, let's change the value bounds. Since we know we have to burn prograde to get into a circular orbit, set the upper bound to -1000 m/s (forces a major retrograde solution). Change the lower bound to -3000 m/s so the optimizer has some breathing room. The normal component is something that will vary depending on how your optimized solution is looking at this point, so set that to -2000 m/s and 2000 m/s for lower and upper bounds, respectively. And radial delta-v is terribly inefficient, so let's change that to -50 m/s and 50 m/s to limit the optimizer in that respect.

You should have something like this:

Edit DV Maneuver					
Maneuver Name					
Laythe Injection					
Proscribed Delta-V (Orbital Vector) ▼					
LV-909 ▼					
Delta-V Vector					
Prograde	-2000	m/s	<input checked="" type="checkbox"/> Opt?	-3000	-1000
Normal	0	m/s	<input checked="" type="checkbox"/> Opt?	-2000	2000
Radial	0	m/s	<input checked="" type="checkbox"/> Opt?	-50	50
Save & Close Cancel					

Figure 20: Laythe Injection Burn

Tap "Save and Close."

Step 10: Optimizing to Minimize Eccentricity

Bring up the optimizer window with control-O (you're using the keyboard shortcut by now, I hope).

Go ahead and remove those active constraints we have in there. Since we deactivated the variables on the TJI burn, we can't influence those anymore. Use the left array in the Constraints box to remove them one by one.

Now, let's change the objective function. On the left, swap the Objective Function to "Minimize Eccentricity." Change the Applicable Event to "Event 5: Laythe Injection." Keep the Applicable Body at "Laythe."

At this point, you should have:

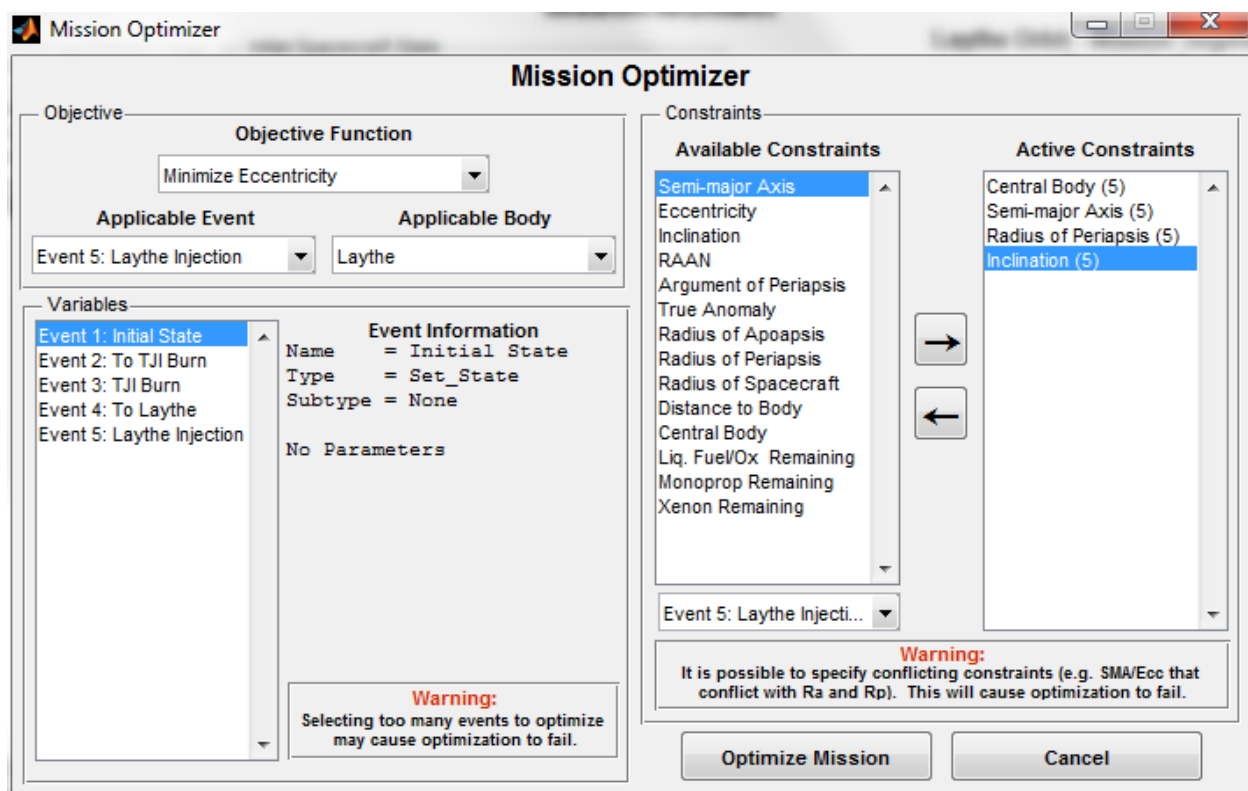


Figure 21: Optimizer Round 3

Also! Make sure to go through the variables list box, select each event, and make sure that only "Event 5: Laythe Injection" has a little "(Opt)" flag next to each of the parameters of the burn. In particular, events 2 and 3 should not have the "(Opt)" flag anymore.

Now let's add some constraints. First, change the dropdown box under the Available Coasts box to "Event 5: Laythe Injection." Then re-add the Central Body constraint as you did before, using Laythe:

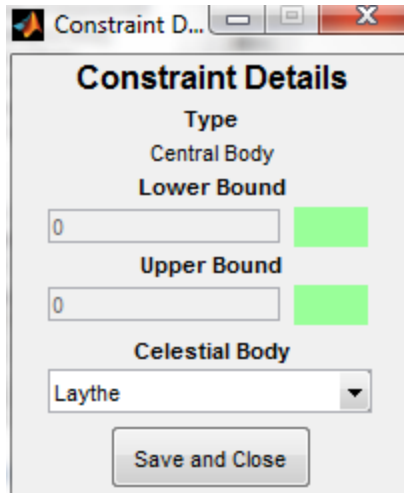


Figure 22: Central Body Constraint

Your optimization window should look like this when you're done:

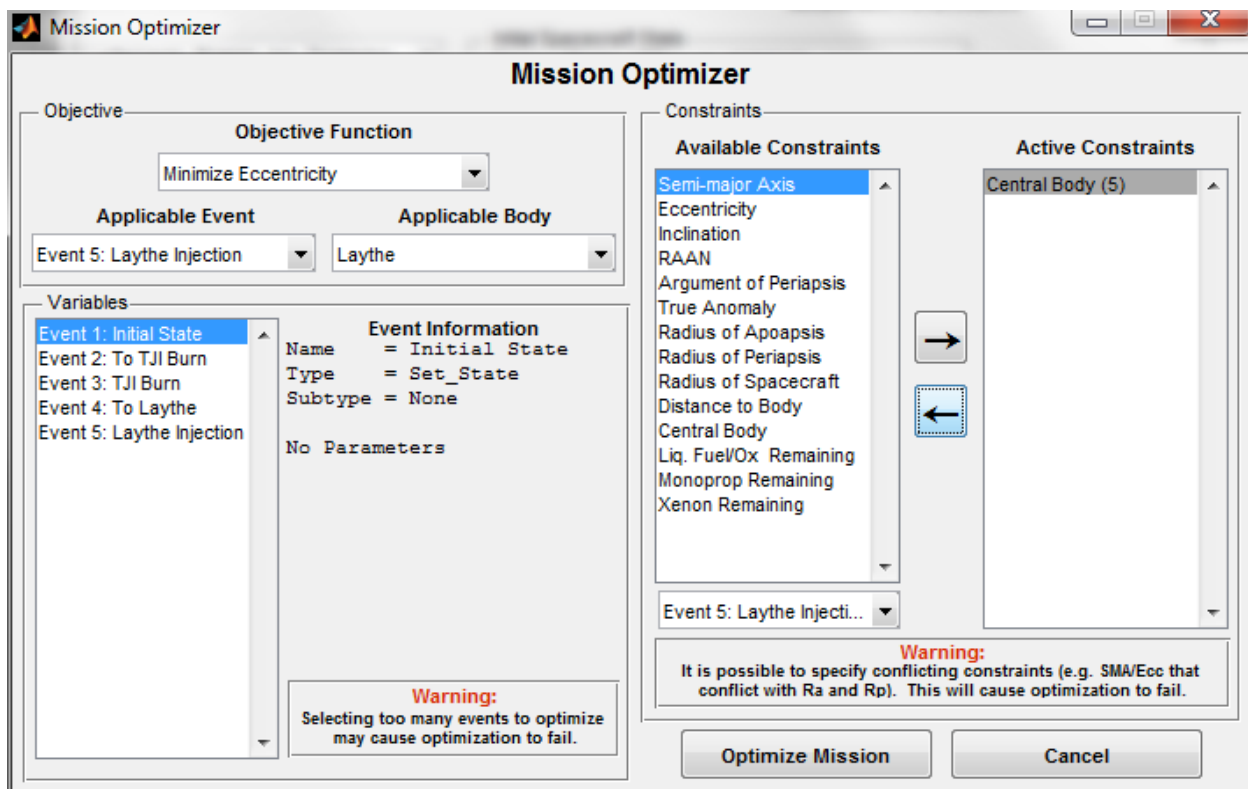


Figure 23: Optimization Round 3

Tap “Optimize Mission” and start praying again.

Here’s an interesting note while you wait: if the optimizer never looks like it’s making moves towards the solution, it’s probably one of two problems:

1. You're up against a variable boundary. Widen up the bounds on your variables in the mission script.
2. You've started very infeasible (meaning a constraint was violated with initial guess).

The second point is most important. If you start with a constraint very violated, the optimizer may get confused and not know how to make progress towards the goal. The optimizer always tries to find feasible solutions, and if it can't, it won't make much progress on optimizing what you're looking for.

If you suspect this is happening, start removing constraints and widening bounds on your variables. Only do one constraint or boundary at a time so you can study how changing the bounds allows the optimizer to find a solution.

Wait for the optimizer to finish, or when you see the eccentricity is small enough, stop the optimizer yourself.

Step 11: Optimizing to Minimize Inclination

Once the eccentricity optimization is complete, let's move to getting that inclination down!

Bring up the Mission Optimization dialog box with control-O again. Let's add a constraint for eccentricity now, especially since our current eccentricity is very close to 0 after our last optimizer run.

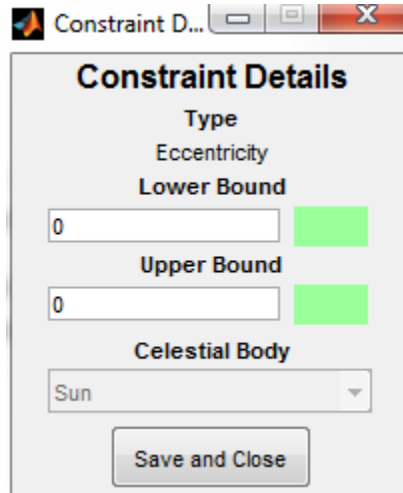


Figure 24: Eccentricity Constraint

Change the objective function to “Minimize Inclination.” Your screen should look like this:

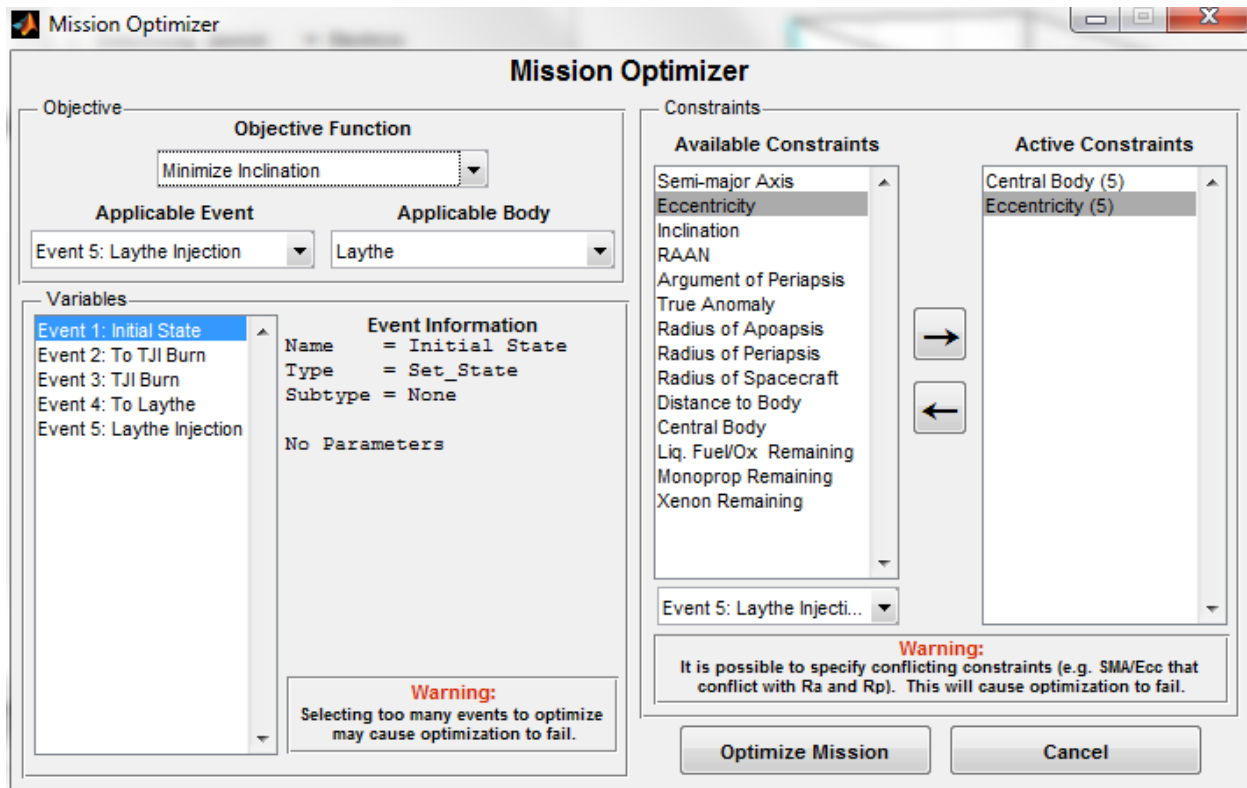


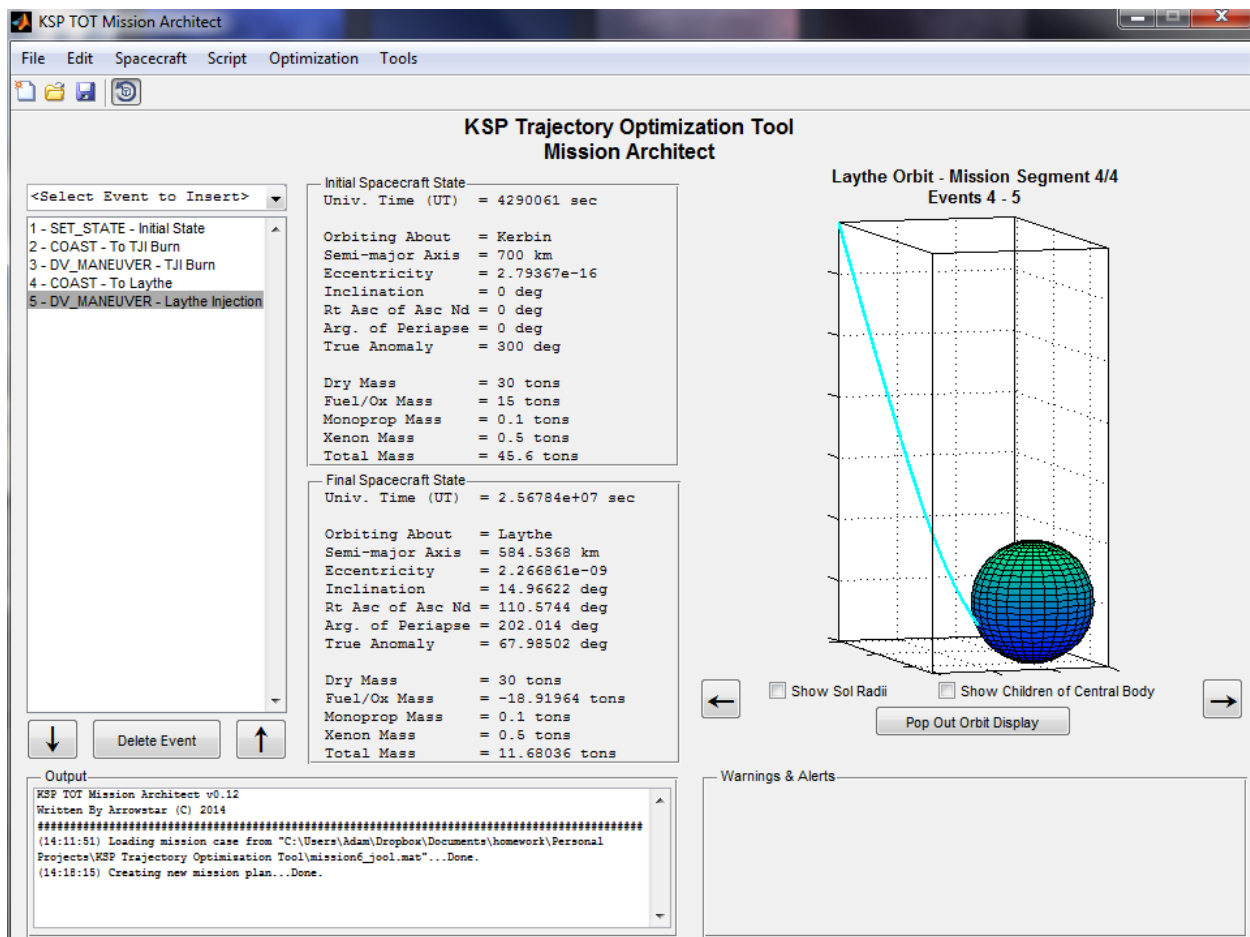
Figure 25: Optimization Round 4

Tap “Optimize Mission.”

Notice what we did here. We optimized eccentricity down to 0.0, and then set a constraint on an eccentricity of 0.0. This way, the optimizer starts with a feasible solution and it won’t be subject to the problems that come with starting infeasible.

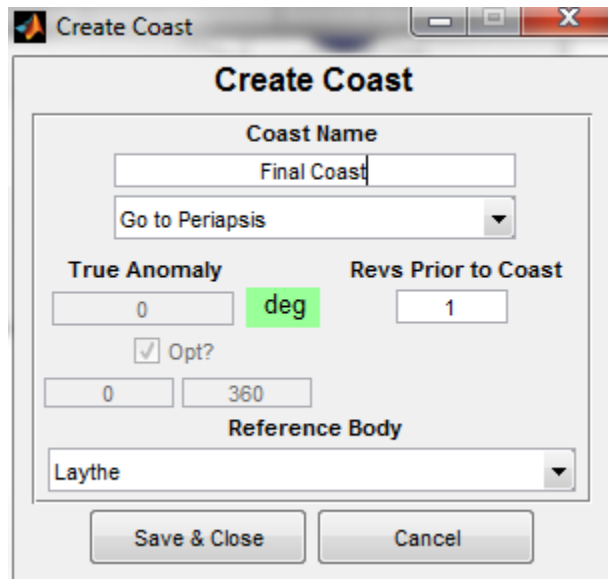
This technique of optimizing a quantity and then putting a constraint on that quantity when you optimize another quantity to keep the first quantity at its current value is a good one. Note that if you wanted an eccentricity other than zero, you could have set a constraint on eccentricity to, say, 0.2 (lower) and 1.0 (upper). The optimizer will minimize the eccentricity down to 0.2 and then stop.

After minimizing eccentricity, my mission looks like this



Step 12: Add Final Coast

Let's add a coast to the mission script after our Laythe injection burn. Make it look like this:



Create Coast

Coast Name
Final Coast

Go to Periapsis

True Anomaly
0

Revs Prior to Coast
1

deg

☒ Opt?

0 360

Reference Body
Laythe

Save & Close Cancel

Figure 26: Final Coast

The one revolution prior to coast makes the spacecraft go around Laythe once before going to periapse. Now view the orbit visualization and look at the impact of the burn we added and optimized:

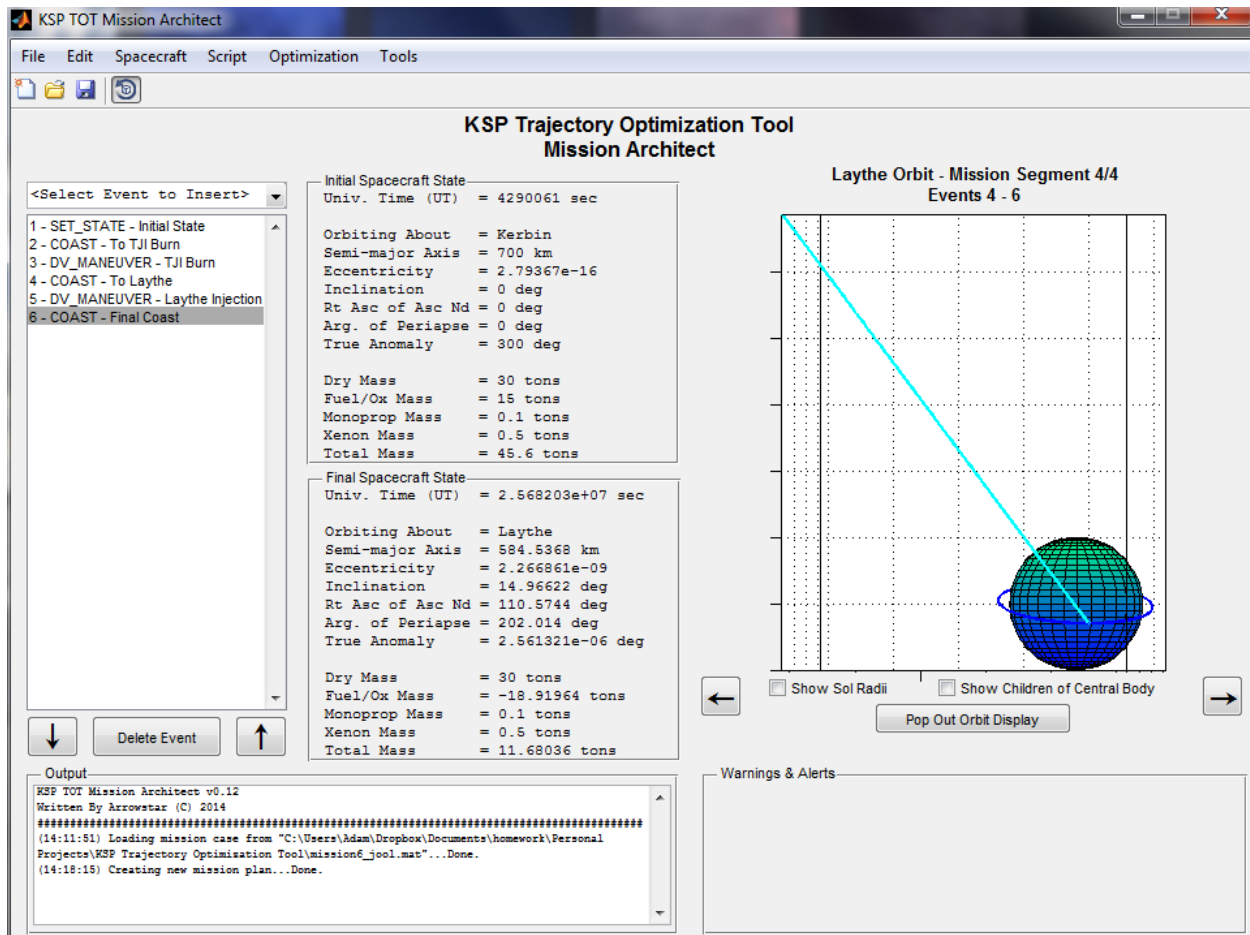


Figure 27: Final Mission

Notice we didn't quite get inclination down to 0.0 degrees. To clean this up, you could add another coast to the ascending node and then repeat the process for minimizing inclination. Or, instead of the "To Laythe" coast going to Laythe periapse, it could go to the Laythe ascending/descending node and you could minimize inclination from that point. Either are acceptable solutions to the problem.

Final Thoughts

There's a multi-step process we went through here to get to Laythe. It can be expanded to go anywhere. In short:

1. Use the optimizer to target the body you want to get to.
2. Use the optimizer to set the periapse distance to something appropriate (say, above the atmosphere).
3. Use the optimizer to set each orbital parameter in turn. You can set SMA, eccentricity, and inclination using three of the four objective functions provided.

Remember to only add constraints when you truly need them, and make sure when you optimize, your initial guess is feasible (that is, your current mission plan mostly falls into the constraints).

In any event, that's it! Congrats, you've just used Mission Architect to figure out how to get from Kerbin to Laythe. Please let me know if you've encountered any problems with this tutorial or with Mission Architect itself by posting on the KSP Trajectory Optimization Tool thread on the KSP Forums.

Thanks, happy orbiting, and Hail Probe!

-Arrowstar