LT Combined Maneuvers via the Alfano Method r1.2

Preprint · September 2019

1 author:

Clyde Colin Helms
Freelance Rocket Science

Implementing Low Thrust Combined Maneuvers via the Alfano Method

C. Colin Helms

Freelance Rocket Science

Author Note

This work is draft 1.2,

23 Jun 2019

Abstract

This paper revisits the algorithm for long duration constant thrust circle to circle transfer first proposed in Edelbaum 1961 [1], as restated by Alfano and Wiesel, 1985 [2]. The method varies from Alfano and Wiesel in that it directly computes the control parameter, generating a table of controls as a function of orbit ratio over the range of Hamiltonian costates.

Alfano's derivation and result is included as an example of a low thrust control law in the textbook, Fundamentals of Astrodynamics and Applications by David A. Vallado [3] (pp. 382-388) and is important due to its influence on astrodynamics students. The treatment in the original paper is abbreviated neither provides an intuitive basis nor describes important features in the derivation; for instance, the control law is limited to combined inclination change less than 45 degrees, Alfano and Wiesel simply tack on additional inclination change for those maneuvers in excess of the angular thrust developed by the control parameters.

The development returns to the original Edelbaum paper to garner additional insights necessary for computation of the control parameter and justification of the algorithm. An ephemeris model is developed to compare results with to the Alfano and Wiesel Nomogram (Figure 1 of Alfano & Wiesel).

Keywords: Low-thrust, Geosynchronous, Orbit Transfer, Electric Propulsion, Inclination

Low Thrust Combined Maneuvers via the Alfano Method

Edelbaum and his team at United Aircraft derived the first closed form solution for continuous thrust circle-to-circle transfers [1]. Alfano and Wiesel extended the method, optimizing the Edelbaum yaw control law for many revolutions [2], other subsequent authors sought to improve the control law [4]. This paper seeks to implement the Alfano solutions in a software tool.

First, we revisit the original derivation with the objective of developing a direct computation of the Edelbaum/Alfano thrust controls.

Initial Development of the Closed Form Solution

The Edelbaum team at United Aircraft started their development from the spherical form of the Lagrange planetary equations. Their 1961 paper surveyed a wide-ranging set of cases for satellite control, both impulsive and continuous thrust. It is worth noting that the Edelbaum development was very general in that the paper developed formulations for combined maneuvers changing any two orbital elements. Optimal combined maneuvers for circle-to-circle transfer using low-thrust propulsion were discussed in an appendix, for cases of change in eccentricity, altitude, and inclination. In this paper we are interested in combined maneuvers changing altitude and inclination.

The Edelbaum development proceeds by combining expressions for thrust vector components pitch (α) and yaw (β) with the Lagrangian equations of motion. With assumptions that time of the maneuver depends only upon the apogee radius (i.e. a circular orbit) and that the inclination change is small, Edelbaum proceeds to derive the steering law for producing the maximum change in inclination and semi-major axis (SMA) over the period of one orbit. The treatment yields the yaw steering angle to achieve the maximal combined altitude increase and inclination change.

The Edelbaum equations of motion are repeated in Equation 1 and Equation 2. These correspond to equations 18 in the original paper, where we have replaced thrust with specific acceleration.

Equation 1

$$\frac{\mathrm{da}}{\mathrm{dt}} = a_0 \frac{A_0}{V_0} (2 \cos \alpha \cos \beta)$$

Equation 2

$$\frac{\mathrm{di}}{\mathrm{dt}} = \frac{A_0}{V_0} [\cos(\theta') \sin \beta]$$

Where:

- a_0, starting semi-major axis
- A_0, is specific acceleration, thrust divided by mass
- V_0, is the initial circular velocity at start of the maneuver
- α, alpha, an in-plane thrust angle, known as pitch
- β, the out-of-plane thrust angle, known as yaw
- θ’, the angle along track, the sum of the argument of periapsis and the true anomaly

The thrust components are formulated in the Velocity-Normal-Binormal reference frame where unit vector $\widehat{x_1}$ is in the direction of the velocity vector, $\widehat{x_2}$ is in the direction of the angular momentum vector, and the binormal, $\widehat{x_3}$ is the cross-product $\widehat{x_1} \times \widehat{x_2}$.

Circular orbits with plane change should measure the along-track angle from the line of nodes, this is known as the Argument of Latitude, shown as $\theta' = \omega + \theta$ in Edelbaum.

.

The thrust angle is the angle measured relative to the Normal axis and can have both in-plane (pitch) and out-of-plane (yaw) components. In-plane components are used for change of eccentricity or semi-major axis, and out-of-plane components affect inclination. The term $\cos\alpha\cos\beta$, provides the tangential component of thrust and the term $\sin\beta$ provides the out-of-plane component of thrust. The expression $\cos(\theta')\sin\beta$ ensures that the resultant out-of-plane thrust is maximum at the line of nodes, and changes sign at the line of apsides.

Equation 3 shows the optimization can proceed by simply maximizing the change in orbital elements with change in pitch and yaw thrust angles and using an adjoint constraint (λ) for the amount if inclination change.

Equation 3

$$\frac{\partial}{\partial\alpha}\frac{\partial}{\partial\beta}\left[\frac{da}{dt} + \lambda\frac{di}{dt}\right] = 0$$

substituting a simple steering law,

$$\tan\beta = k'\cos(\theta')$$

And the following standard trigonometric substitutions,

$$\tan\beta = x$$

$$\cos\beta = \frac{1}{\sqrt{1+x^2}}$$

$$\sin\beta = \frac{x}{\sqrt{1+x^2}}$$

The Edelbaum equations of motion then become,

Equation 4

$$da = {a_0}^2\frac{A_0}{{V_0}^2}\left[\frac{2\sqrt{1-k^2}}{\sqrt{1+k^2\cos^2\theta'}}\right]d\theta'$$

$$di = a_0 \frac{A_0}{{V_0}^2} \left[\frac{k - k\,sin^2\,\theta'}{\sqrt{1 + k^2 \cos^2 \theta'}} \right] d\theta'$$

Where the term under the radical can be recognized as the denominator of the elliptic integral. The control law becomes,

Equation 5

$$\tan \beta = k \cos(\theta')$$

$$k = \frac{k'}{\sqrt{1 + k'^2}}$$

Note that if k is the trigonometric substitution with $k' = sin\beta_{max}$, where β_{max} is the maximum yaw angle for the revolution. When Equations 4 are integrated over one complete orbital period, Equation 6 and Equation 7 result.

Equation 6

$$\Delta a = 8 \frac{{a_0}^3}{\mu} A_0 \left[\sqrt{1 - k} K(k) \right]$$

Equation 7

$$\Delta i = 4 \frac{{a_0}^2}{\mu} A_0 \left[\frac{1}{k} E(k) + \left(k - \frac{1}{k} \right) K(k) \right]$$

Where:

- The complete elliptic integral of the first kind is, K(k).
- The complete elliptic integral of the second kind is, E(k).
- The argument k is the elliptic modulus.
- The square of the circular characteristic velocity ${V_0}^2 = \mu / a_0$,
- a_0 the semi-major axis at start of each revolution.

Using Equation 6 and Equation 7, solutions for k provide the optimum control angle. As the out-of-plane angle varies, inclination change is induced while the given change in semi-major axis is accomplished. Each value of lambda in Equation 3 provides a specific combination of altitude and inclination change for the circle-to-circle transfer in Edelbaum.

Alfano & Wiesel Formulation

Alfano & Wiesel reformulated the Edelbaum equations for large changes in semimajor axis and inclination over many revolutions [2]. Their result selects a different optimum value for substitution into the control law, one which varies for change in semi-major axis per revolution.

Alfano and Wiesel factor the period out of Equation 6 and Equation 7 using,

$$dt = 2\pi\sqrt{\frac{a^3}{\mu}}$$

Equation 8

$$\frac{da}{dt} = \frac{4}{\pi}\sqrt{\frac{a^3}{\mu}}\ A(t)\sqrt{1-u}\,\mathrm{K}(u)$$

Equation 9

$$\frac{di}{dt} = \frac{2}{\pi}\sqrt{\frac{a}{\mu}}A(t)\left[\frac{1}{\sqrt{u}}\mathrm{E}(u) + \left(\sqrt{u} - \frac{1}{\sqrt{u}}\right)\mathrm{K}(u)\right]$$

Alfano and Wiesel have substituted $\sqrt{u}$ = k in the Edelbaum equations, where k is the elliptic modulus and corresponds to the elliptic parameter [5]. This substitution alters the physical interpretation of the argument. Thus, Alfano and Wiesel restate the steering law Equation 5 as,

Equation 10

$$\tan\beta = \frac{\cos\theta'}{\sqrt{{}^{1}/_{u} - 1}}$$

The denominator of Equation 10 is the form of an arctangent of an angle defined by $u = sin(a)$.

Equation 8 and Equation 9 are intended for use over many revolution trajectories and the depletion of fuel affects the acceleration term.

$$A(t) = \frac{A(t_0)}{1 - \dot{m}t}$$

Where the mass rate is, $\dot{m}$.

Alfano makes a differential substitution in order to remove the time dependence of the acceleration.

$$\frac{dA}{dt} = \frac{A(t_0)}{\dot{m}} \ln(1 - \dot{m}t)$$

$$d\tau = A(t)dt$$

Where $d\tau$ is the incremental change in velocity.

By direct substitution Equation 11 and Equation 12 result:

Equation 11

$$\frac{da}{d\tau} = \frac{4}{\pi}\sqrt{\frac{a^3}{\mu}}\,\sqrt{1-u}\,\mathrm{K}(u)$$

Equation 12

$$\frac{di}{d\tau} = \frac{2}{\pi}\sqrt{\frac{a}{\mu}}\left[\frac{1}{\sqrt{u}}\mathrm{E}(u) + \left(\sqrt{u} - \frac{1}{\sqrt{u}}\right)\mathrm{K}(u)\right]$$

For compactness, Alfano bundles the elliptic integrals.

Equation 13

$$P(u) = \sqrt{1-u}K(u)$$

Equation 14

$$R(u) = \frac{1}{\sqrt{u}}E(u) + \left(\sqrt{u} - \frac{1}{\sqrt{u}}\right)K(u)$$

For direct calculation, the derivatives of these functions are needed,

Equation 15

$$\frac{dP(u)}{du} = -\frac{1}{2}\frac{K(u)}{\sqrt{1-u}} + (1-u)^{\frac{1}{2}}K'(u)$$

Equation 16

$$\frac{dR(u)}{du} = \frac{K(u)}{\sqrt{u}} + \frac{K(u) - E(u)}{2u^{3/2}} - \frac{E'(u) - K'(u)}{\sqrt{u}} + \sqrt{u}K'(u)$$

The Hamiltonian is formed,

Equation 17

$$H = 1 + \lambda_a \frac{4}{\pi}\sqrt{\frac{a^3}{\mu}}\, P(u) + \lambda_i \frac{2}{\pi}\sqrt{\frac{a}{\mu}} R(u)$$

With the optimality condition:

$$\frac{\partial H}{\partial u} = 0$$

Equation 18

$$\frac{da}{d\tau} = \frac{\partial H}{\partial \lambda_a} = \frac{4}{\pi}\sqrt{\frac{a^3}{\mu}}\, P(u)$$

Equation 19

$$\frac{di}{d\tau} = \frac{\partial H}{\partial \lambda_i} = \frac{2}{\pi}\sqrt{\frac{a}{\mu}} R(u)$$

These equations show that R(u) controls the amount of inclination change per orbit and P(u) controls the amount of change in orbit ratio. The quantity under the radical in Equation 18 is the Keplerian period and the like quantity in Equation 19 is the inverse of the characteristic velocity. This implies that P(u) has units of *1/sec* and R(u) has units of *km/sec*.

We can solve the value of lambda for a boundary condition of final inclination and final orbit ratio using the transversality conditions,

Equation 20

$$\frac{d\lambda_a}{d\tau} = -\frac{\partial H}{\partial a}$$

$$= \lambda_a \frac{6}{\pi}\sqrt{\frac{\mu}{a^3}}P(u) + \lambda_i \frac{2}{\pi}\sqrt{\frac{\mu}{a}}R(u)$$

Equation 21

$$\frac{d\lambda_i}{d\tau} = -\frac{\partial H}{\partial i} = 0$$

Equation 21 results since there is no dependency on inclination in the Hamiltonian, showing that λ_i is a constant. In addition, the Hamiltonian has no explicit dependency on τ.

$$\frac{\partial H}{\partial \tau} = 0$$

Therefore, the Hamiltonian itself is a constant, and when the spacecraft reaches its final orbit, we may set $\mathrm{H}(\tau_f) = 0$.

Now using simple substitution, λ_i can be solved in terms of λ_a.

$$\lambda_a = -\frac{\lambda_i}{2a}\frac{R'(u)}{P'(u)}$$

$$H = 1 + \lambda_i \frac{2}{\pi}\sqrt{\frac{a}{\mu}}\left[R(u) - \frac{R'(u)P(u)}{P'(u)}\right] = 0$$

Equation 22

$$\lambda_i = \frac{\pi}{2}\sqrt{\frac{\mu}{a}}\frac{-1}{\left[R(u) - \frac{R'(u)P(u)}{P'(u)}\right]}$$

Equation 22 is used to find the constant value of λ_i for the transfer, where the values $\sqrt{u}$ = k, the elliptic modulus in Equation 4.

Alfano simplifies Equation 22 by defining yet another function of u (note the sign change),

Equation 23

$$\phi(u) = \left[\frac{R'(u)P(u)}{P'(u)} - R(u)\right]$$

$$\lambda_i = \frac{\pi}{2}\sqrt{\frac{\mu}{a}}\frac{1}{\phi(u)}$$

Alfano states that the $\phi(u)$ function is monotonic, and its inverse function can be found, which given that λ_i is constant, will solve for the values of u proportional to the square root of orbit ratio, a. One solution, given in the Appendix to [2] is to approximate the inverse $\phi(u)$ using a Chebyshev polynomial. We choose a different approach.

The Steering Law

By substituting the relationship of u to the elliptic modulus, $\sqrt{u} = k$ and using the fundamental trigonometric identity,

$$\mathrm{k} = \sin b$$

$$\tan b = \frac{k}{\sqrt{1-k^2}};$$

The steering law in Equation 10 can be re-arranged.

Equation 24

$$\tan\beta = \frac{\sqrt{u}}{\sqrt{1-u}}\cos\theta' = \frac{k}{\sqrt{1-k^2}}\cos\theta' = \tan(k)\cos\theta'$$

We see that $k = \sqrt{u}$ is the sine of the maximum yaw angle in each orbit. The difference between the Edelbaum steering law, Equation 4, and the Alfano steering law is a Tangent of the argument versus a Sine in Edelbaum. Physically this means that the Alfano control law provides a maximum thrust angle of 57 degrees, whereas the Edelbaum control law may develop a maximum thrust angle of 88 degrees.

In the Edelbaum formulation, the transfer takes place between an initial circular orbit and a final circular orbit. Thus, there is a single pseudo-optimum steering angle over the entire transfer. In Alfano there is the assumption of many circle-to-circle transfers, and though the adjoint constrain for inclination, λ_i is also found to be constant, each circle-to-circle transfer has a different steering angle. The Alfano approach increases sophistication by optimizing a Hamiltonian formulation with adjoint constraints in both SMA and inclination. This means that the elliptic modulus, $k^2 = u$, in Equation 8 and Equation 9 has different solutions between Edelbaum and Alfano, yet still it is clear that $k = \sqrt{u}$ identifies an angle which defines the optimum thrust vector. In Alfano and Wiesel, the $\phi(u)$ must change in proportion to $a^{-1/2}$, and ${\lambda_i}^{-1}$is the proportionality constant.

$$\phi(u) = \frac{\pi}{2}\sqrt{\mu}\left(\frac{1}{\lambda_i\sqrt{a}}\right)$$

Finding the Trajectory Directly

The quantity needed to control the thrust vector is u, the argument of $\phi(u)$. This suggests simply calculating ϕ for a series of linearly spaced u values. It is only necessary to compute the array of $\phi(u)$ once; these values correspond to the values of λ_i for orbit ratio =1 and are referred to as canonical. The complete range of values of λ_i may be computed by multiplying with the inverse of the orbit ratio. Given the vector of computed λ_i these may be stored in a table and arrays of u created by sorting on the canonical values of λ_i. The sorted table effectively represents the inverse phi function.

Computing $\phi(u)$ involves computing the complete elliptic integrals of the first and second kind, which may be done with the python SciPy special library[1].

```
from scipy import special
special.ellipk(u)
special.ellipe(u)
```

[1] Wolfram Mathematica also provides these functions and is used as a check on results.

Figure 1 shows the result of this computation for u = 0.1 to 1. The starting value of 0.1 is used to avoid another singularity in the elliptic integrals, which skews the plot magnitude.

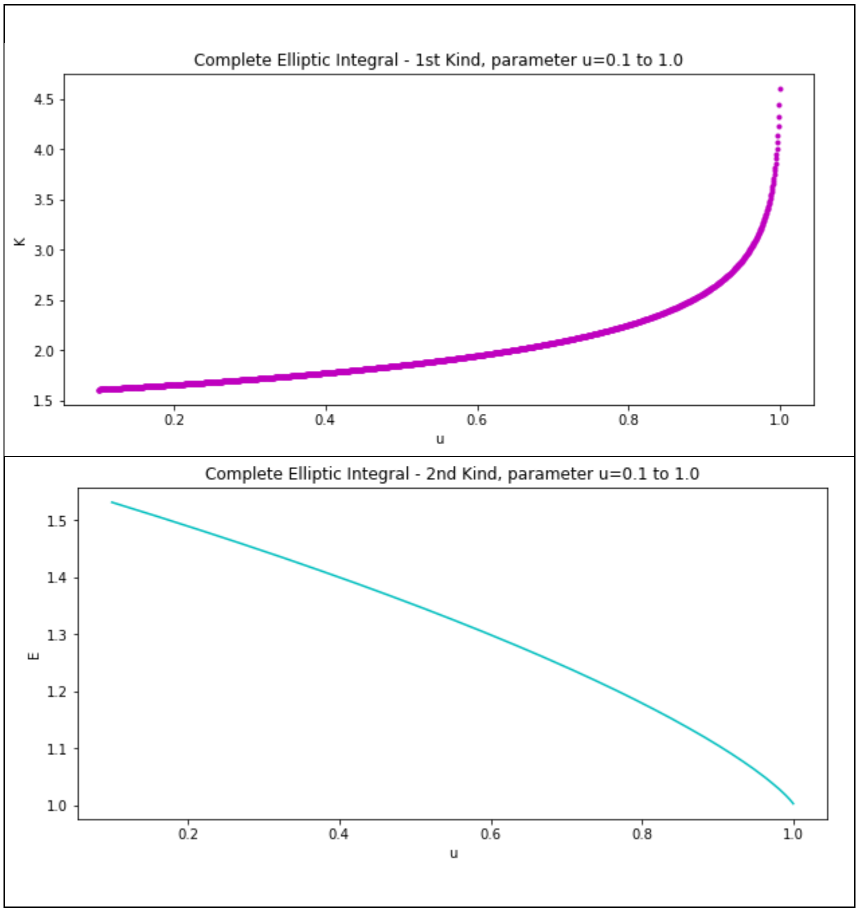


Figure 1, Plot of Complete Elliptic Integrals for u from 0 to 1

The Φ function defined by Alfano also requires that the derivatives of the first and second complete elliptic integral be taken.

Formulas for elliptic integral derivatives are found in the NIST Downloadable Library of Math Functions (DLMF) [6]. The form of the derivatives for K and E in the DLMF use the elliptic modulus as an argument, thus these formulas must be multiplied by ½, which results from taking the derivative of $\sqrt{u}$. The formulas used are as follows.

Equation 25

$$\frac{dK(u)}{du} = \frac{1}{2}\frac{E(u) - (1-u)K(u)}{u(1-u)}$$

Equation 26

$$\frac{dE(u)}{du} = \frac{1}{2}\frac{E(u) - K(u)}{u}$$

Equation 25 and Equation 26 can be used to compute $P'(u)$ and $R'(u)$.

Equation 27

$$P(u) = \sqrt{1-u}K(u)$$

$$\frac{dP(u)}{du} = -\frac{1}{2}\frac{K(u)}{\sqrt{1-u}} + (1-u)^{\frac{1}{2}}K'(u)$$

$$= \frac{E(u) - K(u)}{2u\sqrt{1-u}}$$

Equation 28

$$R(u) = \frac{1}{\sqrt{u}}E(u) + \left(\sqrt{u} - \frac{1}{\sqrt{u}}\right)K(u)$$

$$\frac{dR(u)}{du} = \frac{K(u)}{\sqrt{u}} + \frac{K(u) - E(u)}{2u^{3/2}} - \frac{E'(u) - K'(u)}{\sqrt{u}} + \sqrt{u}K'(u)$$

$$= -\frac{E(u) - K(u)}{2u^{3/2}}$$

Plots of these functions are shown in Figure 2 and Figure 3.

Equation 27 and Equation 28 can be used to compute $\phi(u)$ for a series of u from 0 to 1 per Equation 23. A linear array of u is generated to four decimal place precision using the numpy linspace library function.

```
u = np.round(0.1 * np.linspace(1, 10, ncols, endpoint=False), 4)
```

We exclude the endpoint in this array because the singularity in the recurring expression (1 - u) causes scaling difficulties. The resulting plot of Φ(u) is shown in Figure 4.

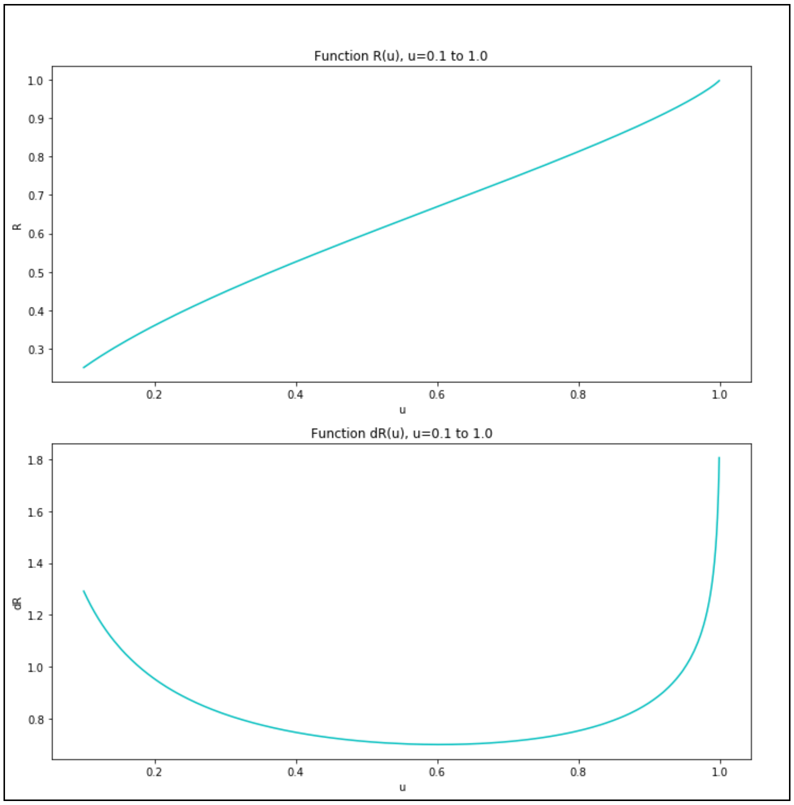


Figure 2, Plots of R(u) and dR/du

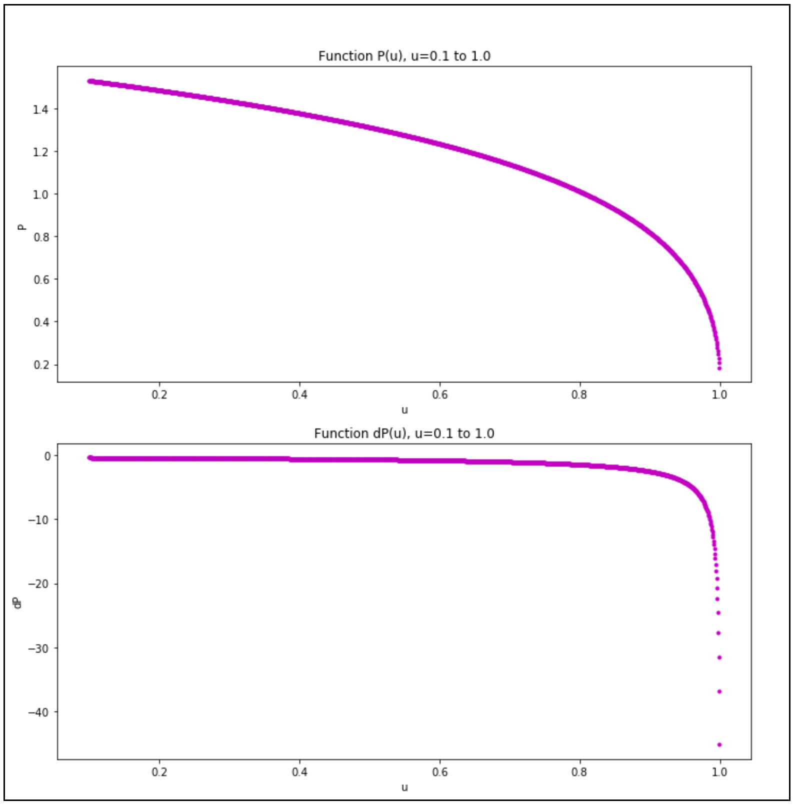


Figure 3, Plots of P(u) and dP/du

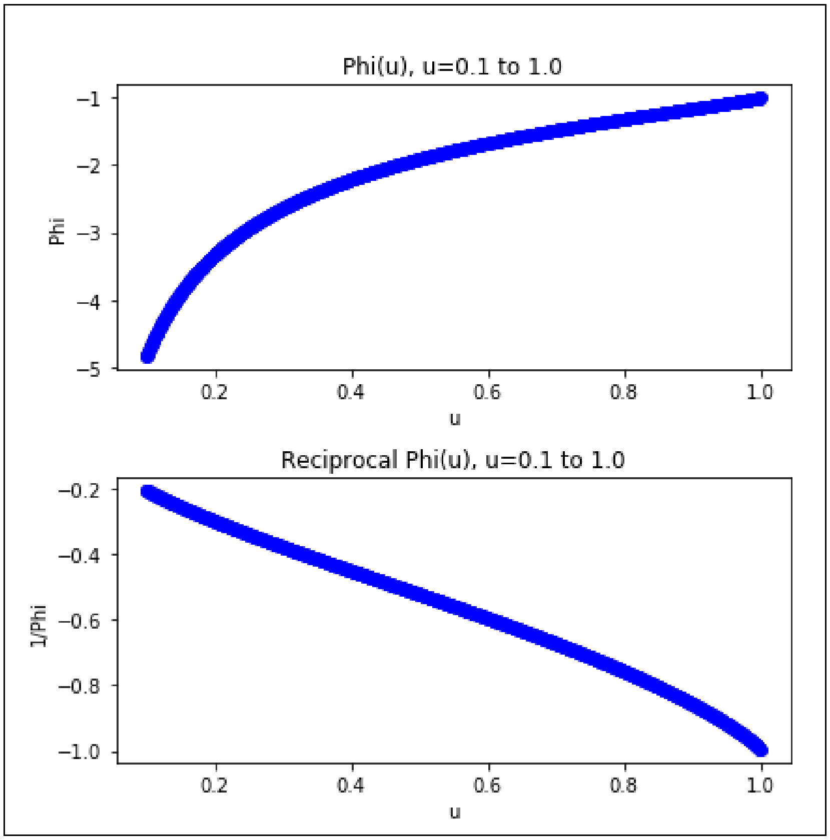


Figure 4, Inverse Φ(u) over the Domain of u

The values of the λ_i costates are formed as combinations of the reciprocal of Φ(u) and the reciprocal square root of the orbit ratio and is shown in Figure 5.

The trajectory for any given value of λ_i can be visualized as a plane parallel to the u, R axes cutting through the Φ surface at a vertical offset equal to λ_i. Starting at a $\lambda_i$value of -0.496 Figure 5 shows the trajectory is cutoff at the right edge, where the u value approaches 1. This agrees with Alfano and Wiesel's original figure as reprinted in Vallado Figure 6-24 [3], which shows that trajectories for $\lambda_i < -0.5$ are predominately inclination change.

Another observation is that for values of λ_i more positive than -0.3245 the trajectory is cutoff at the left edge where the u values are less than 0.2. This indicates that the optimal trajectory does not start until an orbit ratio > 1 has been achieved. This agrees with Vallado Figure 6-24 which shows trajectories less than this λ_i are predominately altitude change.

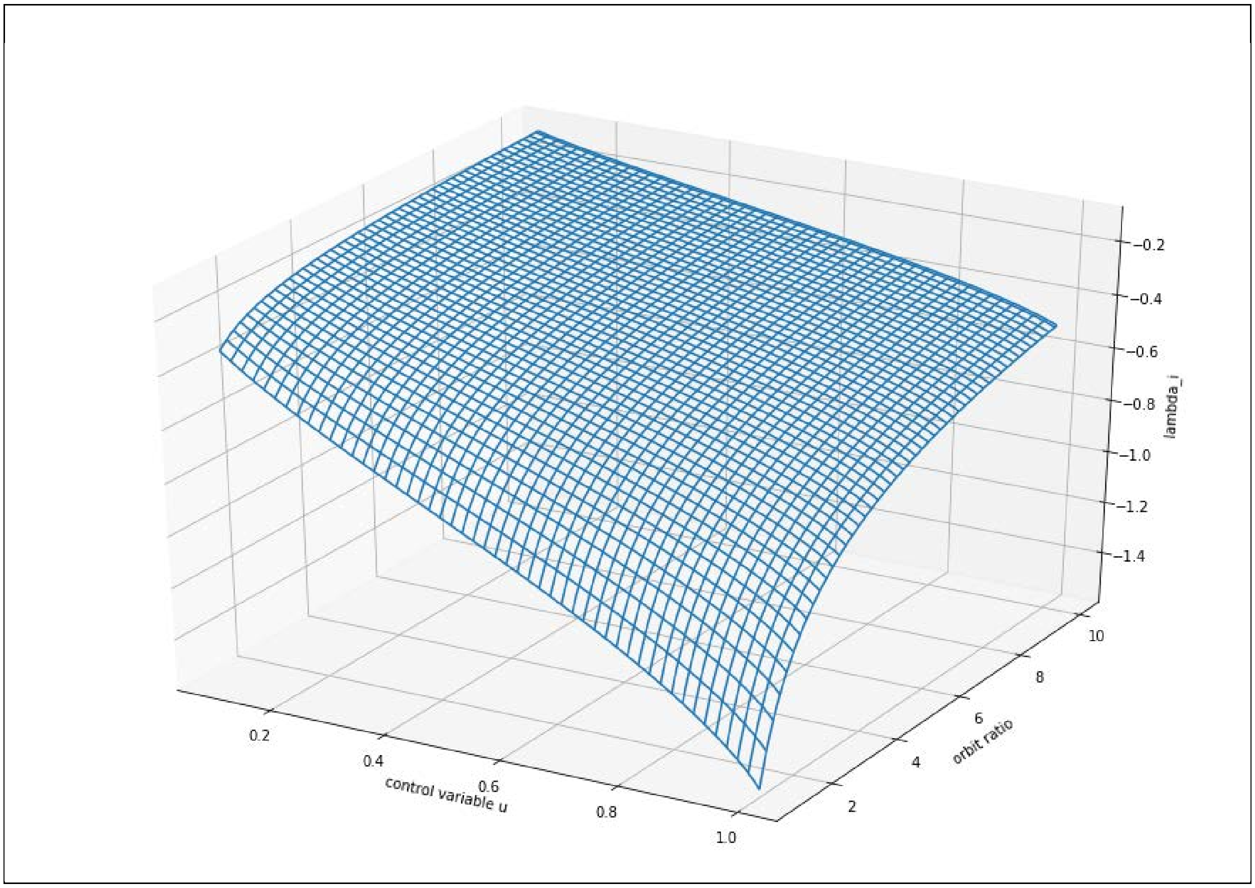
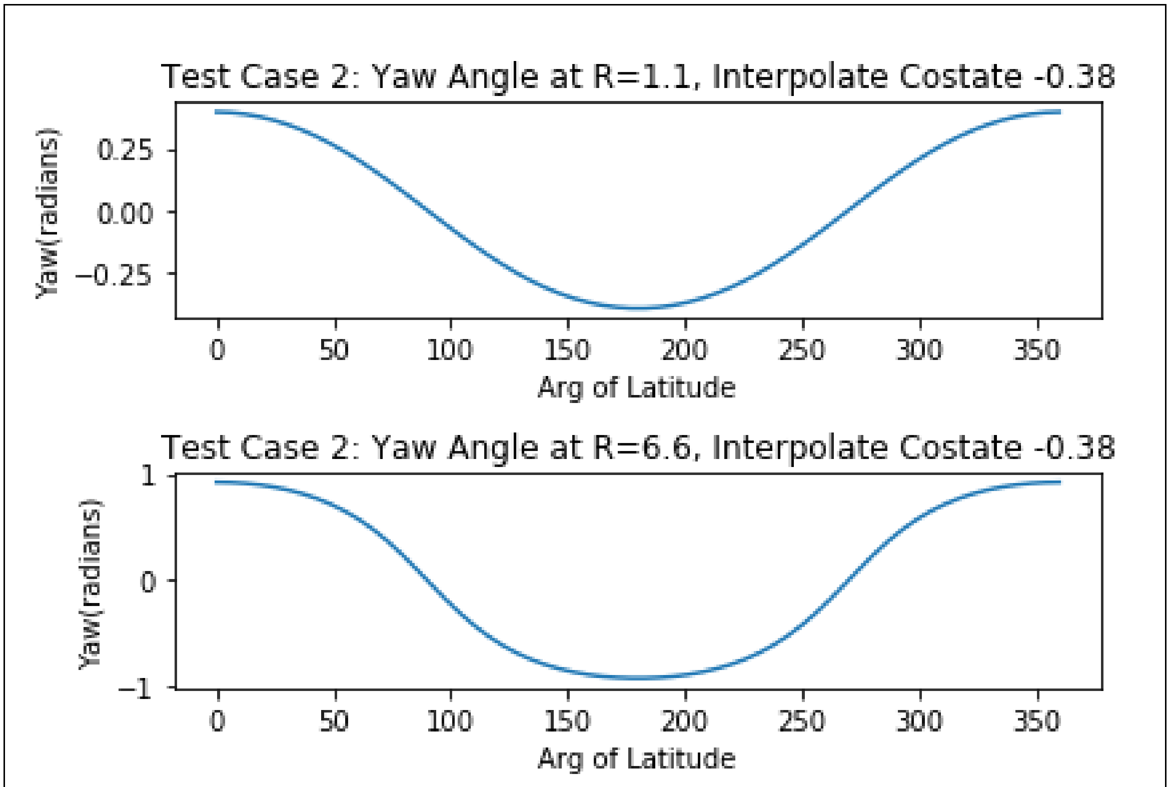
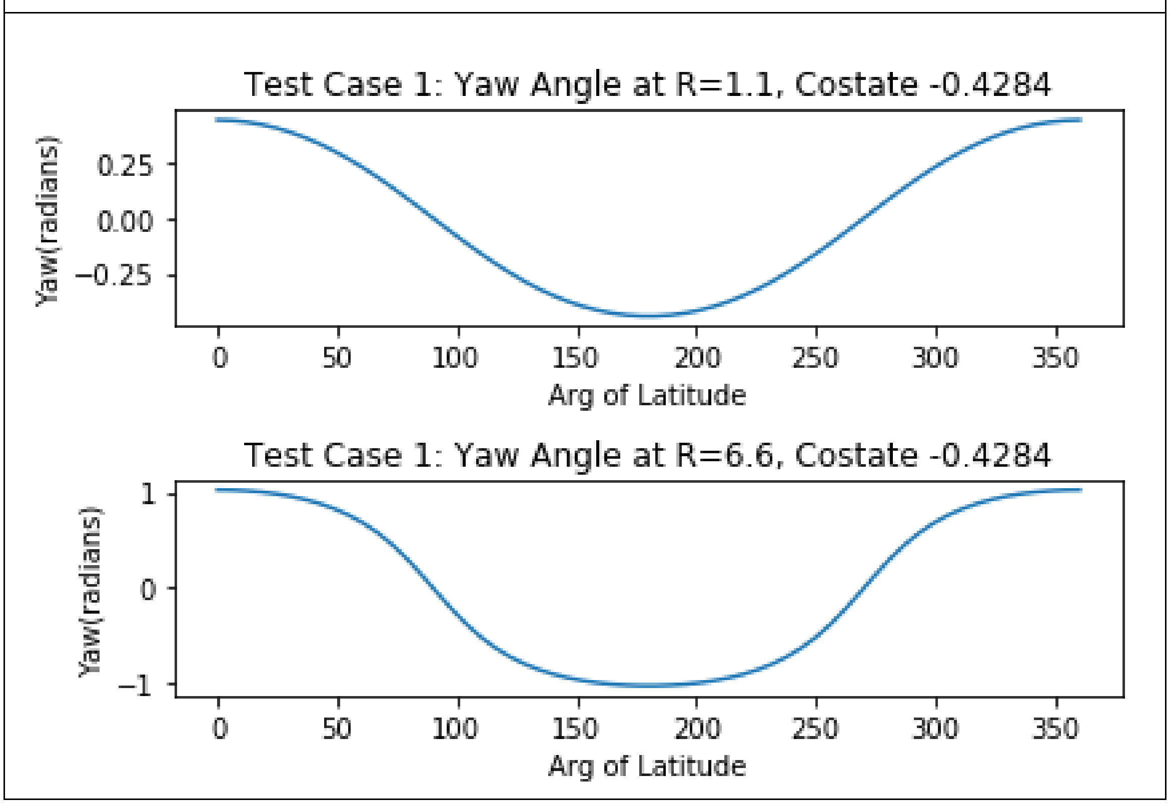


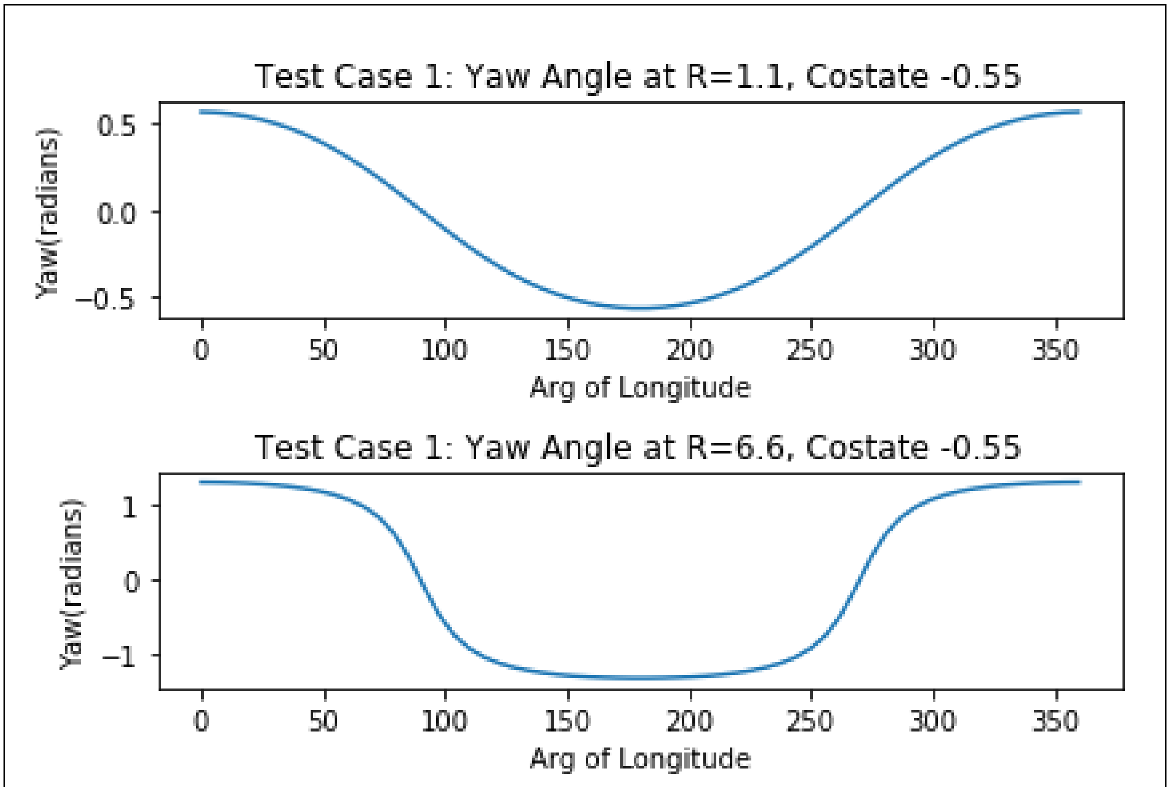
Figure 5, Alfano Costates by Orbit Ratio

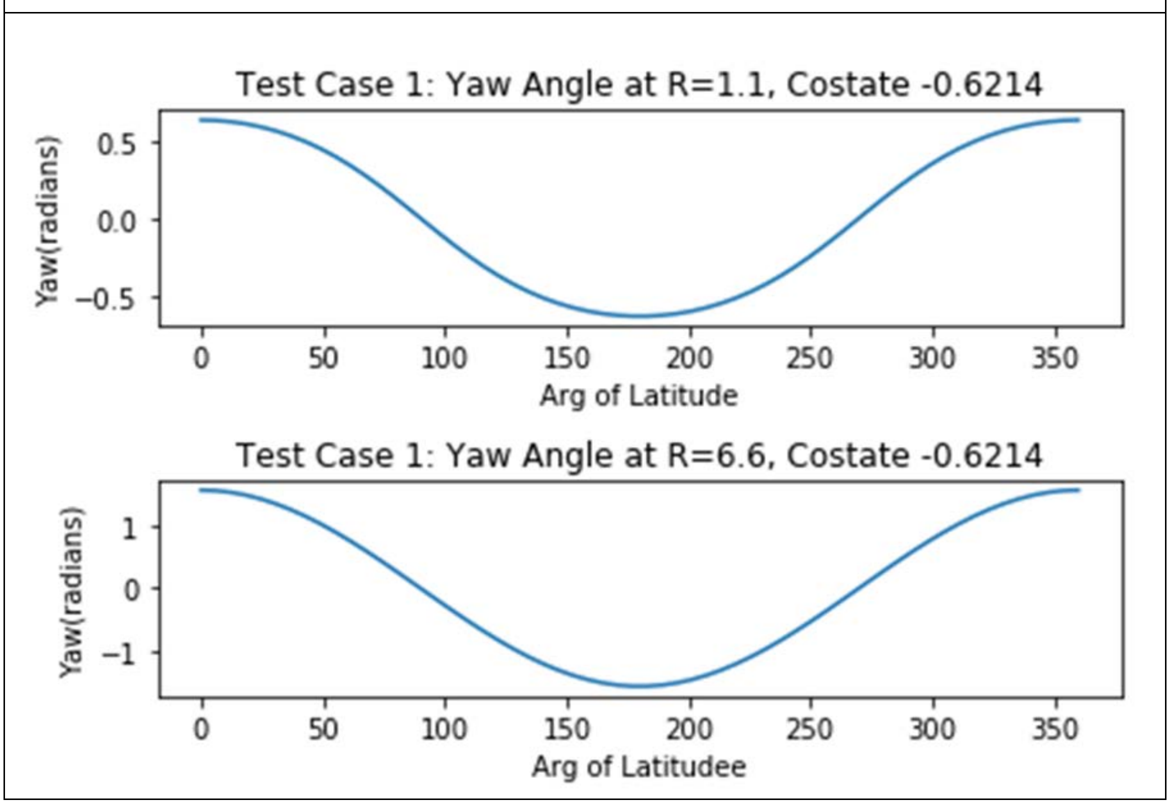
In our method, the end of the trajectory is easily identified because the number of u values associated with any canonical value of λ_i that are more negative than -0.496 will simply run out at some value of orbit ratio less than 10. We mime the behavior of the Alfano and Wiesel in our code by initializing to 1 the rows of the table in which we store the values of u. The table is then overwritten with calculated values of u, leaving the 1 value as residue where no u value is associated with the costate. With this artifice, the control program simply returns a yaw angle of $\pi/2 * \cos\theta'$ when it encounters values of $u = 1$ in the control table.

The control program using this approach provides yaw angles over the range of orbit ratio 1.1 to 6.6 for various values of costate as shown in Figure 6.


Test Case 2: Yaw Angle at R=1.1, Interpolate Costate -0.38
Yaw(radians)
0.25
0.00
−0.25
0 50 100 150 200 250 300 350
Arg of Latitude
Test Case 2: Yaw Angle at R=6.6, Interpolate Costate -0.38
Yaw(radians)
1
0
−1
0 50 100 150 200 250 300 350
Arg of Latitude


Test Case 1: Yaw Angle at R=1.1, Costate -0.4284
Yaw(radians)
0.25
0.00
−0.25
0 50 100 150 200 250 300 350
Arg of Latitude
Test Case 1: Yaw Angle at R=6.6, Costate -0.4284
Yaw(radians)
1
0
−1
0 50 100 150 200 250 300 350
Arg of Latitude


Test Case 1: Yaw Angle at R=1.1, Costate -0.55
Yaw(radians)
0.5
0.0
−0.5
0
50
100
150
200
250
300
350
Arg of Longitude
Test Case 1: Yaw Angle at R=6.6, Costate -0.55
Yaw(radians)
1
0
−1
0
50
100
150
200
250
300
350
Arg of Longitude


Test Case 1: Yaw Angle at R=1.1, Costate -0.6214
Yaw(radians)
0.5
0.0
−0.5
0
50
100
150
200
250
300
350
Arg of Latitude
Test Case 1: Yaw Angle at R=6.6, Costate -0.6214
Yaw(radians)
1
0
−1
0
50
100
150
200
250
300
350
Arg of Latitudee

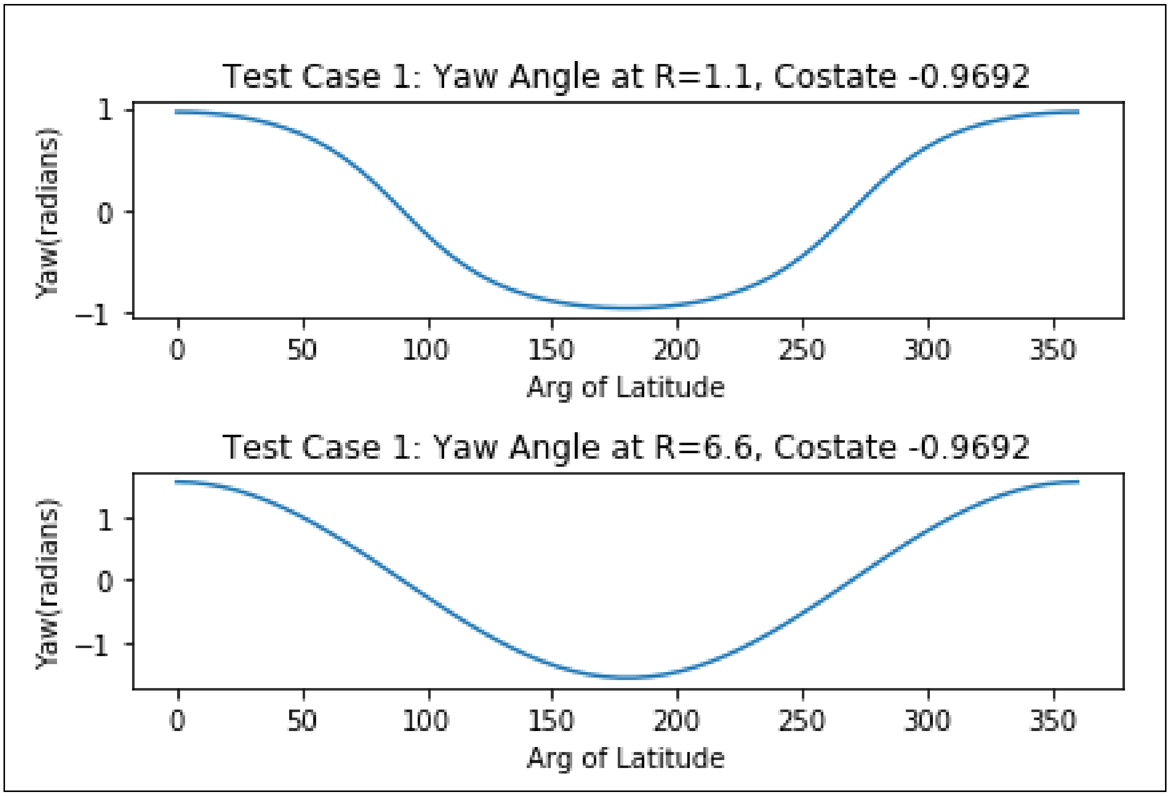


Figure 6, Steering Angles from Costates

The approach for validation of the direct computation is to execute multiple trajectory simulations using the Goddard Mission Analysis Toolkit (GMAT) [7] which provides an Ephemeris model based upon JPL Navigation and Ancillary Information Facility (NAIF) astrodynamics kernels. The GMAT mission model is simple, shown in Figure 6. With the objective of obtaining computed values as similar to Alfano and Wiesel as possible, the eclipse model is turned off and there is no drag or Solar Radiation Pressure model used. The spacecraft model is a 4.5mT, 128kW SEP vehicle using an aggregate of Hall Effect Thrusters providing 6N thrust. Multiple models are executed in batch for orbit ratio 1.5 to 10 in increments of 0.5 and costates from -0.1 to -1.56 in increments of 0.01. The mission is executed starting at 0 degrees inclination, and the resulting final inclination is the maximum achievable orbit ratio and inclination for the given costate. The initial fuel mass is 1855kg which is precalculated to be sufficient for the theoretically maximum delta-v at orbit ratio 10.

```
BeginMissionSequence;

Propagate 'Propagate to periapsis' TrimPropagator(EOTV)
{EOTV.Earth.Periapsis};

BeginFiniteBurn 'GEOTransfer' ContinuousThrust(EOTV);
       GMAT SMA_INIT = EOTV.SMA;
       GMAT REV_LAST = EOTV.Earth.OrbitPeriod;
       GMAT T0_AT_REV = EOTV.ElapsedSecs;
       GMAT REV = REV + 1;

       While EOTV.SMA <= SMA_END

          GMAT AOL = EOTV.TA + EOTV.AOP;
          If AOL > 360.0
             GMAT AOL = AOL - 360.0;
          EndIf;

          GMAT T_REV = EOTV.ElapsedSecs - T0_AT_REV;

          If T_REV >= REV_LAST

             GMAT REV_ERR = T_REV - REV_LAST;
             GMAT REV_LAST = EOTV.Earth.OrbitPeriod;
             GMAT T0_AT_REV = EOTV.ElapsedSecs - REV_ERR;
             GMAT REV = REV + 1;

             GMAT [CONTROL] = Python.YawAngles.get_control_onrev\
             (COSTATE, AOL, EOTV.Earth.SMA, SMA_INIT, MORE);

          EndIf;

          GMAT EOTV.HET1.ThrustDirection1 = CONTROL(1,1);
          GMAT EOTV.HET1.ThrustDirection2 = CONTROL(1,2);
          GMAT EOTV.HET1.ThrustDirection3 = CONTROL(1,3);

          Propagate 'Propagate Steps' DefaultProp(EOTV);

       EndWhile;

EndFiniteBurn 'GEOTransfer' ContinuousThrust(EOTV);
```

Figure 7, Mission Model

A JSON control table containing rows of u values in order of orbit ratio and organized by keys of canonical λ_i is populated by a Python script which computes the elliptic integrals and their derivatives. A slice of the control table is shown as Table 1. Note that where the left edge of the plot occurs in Figure 5, the default value of 1 is shown.

Table 1, Summary of Control Table Output

Orbit R	**Costate Values**									
	-0.3245	**-0.3255**	…	**-0.4952**	**-0.496**	…	**-1.5592**	**-1.5616**	**-1.5637**	**-1.5659**
1	0.1	0.1006	…	0.2181	0.2187	…	0.9969	0.9976	0.9982	0.9988
1.01	0.101	0.1015	…	0.22	0.2206	…	0.9991	1	1	1

1.02	0.1019	0.1025	…	0.2219	0.2225	…	1	1	1	1
1.03	0.1028	0.1034	…	0.2239	0.2245	…	1	1	1	1
…	…	…	…	…	…	…	…	…	…	…
9.97	0.6729	0.6755	…	0.9982	0.9989	…	1	1	1	1
9.98	0.6733	0.6759	…	0.9984	0.9991	…	1	1	1	1
9.99	0.6739	0.6763	…	0.9986	0.9992	…	1	1	1	1
10	0.6742	0.6768	…	0.9988	1	…	1	1	1	1

In the mission sequence, Figure 6, at line 25, a call to Python is made at the beginning of each orbit period. This call reads the *u* value from the JSON control table represented by Table 1, and computes both the yaw and pitch thrust vector per Equation 10. The GMAT variable CONTROL(1,1) is the returned pitch component of the thrust vector, and the returned yaw component is CONTROL(1,2). The CONTROL(1,3) element would be used for a roll component should it be necessary to compensate for solar beta, but is clamped to zero: the default solar power model in GMAT is simple and does not take solar beta angle into account.

Comparison to the Edelbaum Control Law

Given the unexplained difference in control laws between Edelbaum and Alfano, the same mission simulation is performed with the Edelbaum control law,

Equation 29

$$\tan\beta = \frac{k}{\sqrt{1+k^2}}\cos\theta'$$

where, $k = \sqrt{u}$.

For this case, the mission script in Figure 7 is unchanged, however Python code adjusts the u values from the control table as the square root of the stored values and the return control angles are modified to use Equation 29.

(coding in progress – to be completed)

Works Cited

[1] T. N. Edelbaum, "Propulsion Requirements for Controllable Satellites," *ARS Journal,* pp. 1079 - 1089, 1961.

[2] A. a. Wiesel, "Optimal Many-Revolution Orbit Transfer," *Journal of Guidance and Control*, pp. 155-157, 1985.

[3] D. A. Vallado, Fundamentals of Astrodynamcis and Applications, 4th ed., Hawthorne, CA: Microcosm Press, 2013, pp. 381 - 388.

[4] C. Casolino, "AIAA-2004-5402, Improved Edelbaum Approach to Optimize LEO GEO Low-Thrust Transfers," in *AIAA/AAS Astrodynamics Specialist Conference*, Providence, Rhode Island, 2004.

[5] E. W. Weisstein, "Elliptic Modulus."," 03 Jul 2019. [Online]. Available: http://mathworld.wolfram.com/EllipticModulus.html.

[6] F. W. J. Olver and A. B. O. Daalhuis, "NIST Digital Library of Mathematical Functions.," 15 06 2019. [Online]. Available: htto:///dlmf.nist.gov. [Accessed 22 February 2018].

[7] NASA Goddard Spaceflight Center, "GMAT Central," 22 Nov 2012. [Online]. Available: http://gmatcentral.org/. [Accessed 31 May 2018].

[8] R. A. Broucke, "Low-Thrust Trajectory Optimization in an Inverse Square Force Field," in *AAS Paper 91-159*, Houston, 1991.

[9] E. W. Weisstein, ""Complete Elliptic Integral of the First Kind."," 03 Jul 2019. [Online]. Available: http://mathworld.wolfram.com/CompleteEllipticIntegraloftheFirstKind.html.

[10] E. W. Wesstein, "Elliptic Integral of the Second Kind.," 03 Jul 2019. [Online]. Available: http://mathworld.wolfram.com/EllipticIntegraloftheSecondKind.html.