

Program aeroassist

Aero-assist Trajectory Optimization

This document is the user's manual for a Fortran computer program called `aeroassist` that uses optimal control software distributed by Applied Mathematical Analysis to solve the aero-assist trajectory optimization problem. The trajectory is modeled as a single phase with several types of user-defined initial and final boundary conditions. The software attempts to maximize the speed of the vehicle at atmospheric exit or maximize the orbital plane change during the atmospheric phase of the mission. The type of optimization is selected by the user.

The important features of this scientific simulation are as follows:

- Normalized lift coefficient and bank angle control variables
- 3-DOF flight path equations of motion relative to a spherical, rotating Earth
- U.S. Standard 1976 atmosphere model and fourth-order zonal gravity model
- User-defined aerodynamic characteristics and point-mass vehicle properties
- User-defined path constraints such as heat rate and dynamic pressure

The `AMA_OC` software suite is a *direct transcription method* that can be used to solve a variety of trajectory optimization problems using the following combination of numerical methods:

- collocation and *implicit* integration
- adaptive mesh refinement
- sparse nonlinear programming

Additional information about the mathematical techniques and numerical methods used in the `AMA_OC` software can be found in the book, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming* by John. T. Betts, SIAM, 2010.

The `aeroassist` software consists of Fortran routines that perform the following tasks:

- set algorithm control parameters and call the transcription/optimal control subroutine
- define the problem structure and perform initialization related to scaling, lower and upper bounds, initial conditions, etc.
- compute the *right-hand-side* differential equations
- evaluate any point and path constraints
- display the optimal solution results and create an output file

The `AMA_OC` software will use this information to *automatically* transcribe the user's problem and perform the optimization using a sparse nonlinear programming method. The software allows the user to select the type of collocation method and other important algorithm control parameters.

Program execution

An input file created by the user can be run from the command line or a simple batch file with a statement similar to the following:

```
aeroassist geo2leo_max_speed.in
```

If the software is executed without an input file on the command line, the computer program will display the following information screen and file name prompt:

```
*****
*           Program aeroassist           *
*                                           *
*       aeroassist trajectory            *
*           optimization                  *
*                                           *
*           June 9, 2011                  *
*****

please input the name of the simulation definition file
```

The user should respond to this prompt with the name of a compatible input data file including the filename extension.

Input file format and contents

The `aeroassist` software is “data-driven” by a user-created text file. The following is a typical input file used by this computer program. In the following discussion the actual input file contents are in courier font and all explanations are in *times italic* font.

This data file defines a simulation that maximizes the speed at atmospheric exit while enforcing a heat rate path constraint. The simulation starts in a geosynchronous equatorial orbit (GEO) and finishes in an elliptical low Earth orbit (LEO) with a final orbital inclination of 28.5 degrees. The heat rate path constraint enforces a value ≤ 600 BTU/foot²-second.

Each data item within an input file is preceded by one or more lines of *annotation* text. Do not delete any of these annotation lines or increase or decrease the number of lines reserved for each comment. However, you may change them to reflect your own explanation. The annotation line also includes the correct units and when appropriate, the valid range of the input. ASCII text input is not case sensitive but must be spelled correctly.

The first six lines of any input file are reserved for user comments. These lines are ignored by the software. However the input file must begin with six and only six initial text lines.

```
*****
aeroassist trajectory optimization
simulation definition data file ==> geo2leo_max_speed.in
geo-to-leo w/ plane change and aeroheating Constraints
maximize speed at atmospheric exit - June 9, 2011
*****
```

The first program input is an integer that defines the type of entry interface conditions to use. Please consult the "Problem setup" section later in this document for an explanation of these three program options.

```
*****
initial conditions type
*****
1 = user input of flight path coordinates at entry interface
2 = derived from deorbit maneuver; fixed entry conditions
3 = derived from deorbit maneuver; bounded entry conditions
-----
3
```

The next program input is an integer that defines the type of mission to simulate. Please note that program option 0 (no optimization) will solve the two-point boundary value problem (TPBVP).

```
*****
simulation type
*****
0 = no optimization
1 = maximize final speed
2 = maximize inclination change
-----
1
```

This section allows the user to define the vehicle weight, aerodynamic reference area, the nose radius used in the aero-heating calculations, and other vehicle aerodynamic properties.

```
*****
vehicle weight and aerodynamics characteristics
*****

vehicle weight (pounds)
5000.0

aerodynamic reference area (square feet)
125.84

nose radius (feet)
1.0

drag coefficient at zero angle-of-attack; cd0 (nondimensional)
0.032

drag polar constant k (non-dimensional)
1.4
```

The next set of inputs defines the conditions at the deorbit point in the initial circular orbit. The calendar date and universal time are required in order to transform coordinates.

```
*****
deorbit conditions
*****

calendar date at deorbit maneuver (month, day, year)
1,1,2001

universal time at deorbit maneuver (hours, minutes, seconds)
0,0,0

altitude at deorbit maneuver (nautical miles)
19323.0
```

orbital inclination at deorbit maneuver (degrees)
0.0

right ascension of the ascending node at deorbit maneuver (degrees)
0.0

true anomaly at deorbit maneuver (degrees)
30.0

The following series of data items are reserved for the initial conditions at the entry interface (EI) or point of atmospheric entry. To constrain one or more initial conditions, the user should input identical lower and upper bounds. To free or un-constrain one or more initial states, set the lower and/or upper bounds to 1.0d99. Please note the units and valid data range for each item.

flight conditions and bounds at atmospheric entry

NOTE 1: set upper and lower bounds to
the initial value to constrain or
"fix" a flight condition.

NOTE 2: set bound to 1.0d99 to ignore

calendar date at entry interface (month, day, year)
1,1,2001

universal time at entry interface (hours, minutes, seconds)
0,0,0

initial time (seconds)
0.0

upper bound for initial time (seconds)
0.0

lower bound for initial time (seconds)
0.0

initial altitude (feet)
400000.0

upper bound for initial altitude (feet)
400000.0

lower bound for initial altitude (feet)
400000.0

initial velocity (feet/second)
32268.637

upper bound for initial velocity (feet/second)
35000.0

lower bound for initial velocity (feet/second)
20000.0

initial flight path angle (-90 <= fpa <= +90; degrees)
-4.0

upper bound for initial flight path angle (-90 <= fpa <= +90; degrees)
-0.5

lower bound for initial flight path angle (-90 <= fpa <= +90; degrees)
-10.0

initial flight azimuth (0 <= azimuth <= 360; degrees)
94.0

upper bound for initial flight azimuth (0 <= azimuth <= 360; degrees)
1.0d99

lower bound for initial flight azimuth (0 <= azimuth <= 360; degrees)
1.0d99

initial declination (-90 <= declination <= +90; degrees)
0.0

upper bound for initial declination (-90 <= declination <= +90; degrees)
1.0d99

lower bound for initial declination (-90 <= declination <= +90; degrees)
1.0d99

initial east longitude (0 <= longitude <= 360; degrees)
18.0

upper bound for initial east longitude (0 <= longitude <= 360; degrees)
0.0

lower bound for initial east longitude (0 <= longitude <= 360; degrees)
0.0

The following series of data items allow the user to define the flight conditions at atmospheric exit. To constrain one or more conditions, the user should input identical lower and upper bounds. To free or un-constrain one or more final states, set the lower and/or upper bounds to 1.0d99. Please note the units and valid data range for each item.

flight conditions and bounds at atmospheric exit

NOTE 1: set upper and lower bounds
to the final value to constrain or
"fix" a flight condition.

NOTE 2: set bound to 1.0d99 to ignore

final time (seconds)
750.0

upper bound for final time (seconds)
10000.0

lower bound for final time (seconds)
100.0

final altitude (feet)
400000.0

upper bound for final altitude (feet)
400000.0

lower bound for final altitude (feet)
400000.0

final velocity (feet/second)
24314.43

upper bound for final velocity (feet/second)
35000.0

lower bound for final velocity (feet/second)
1000.0

final flight path angle (-90 <= fpa <= +90; degrees)
1.25

upper bound for final flight path angle (-90 <= fpa <= +90; degrees)
+20.0

lower bound for final flight path angle (-90 <= fpa <= +90; degrees)
-20.0

final flight azimuth (0 <= azimuth <= 360; degrees)
90.0

upper bound for final flight azimuth (0 <= azimuth <= 360; degrees)
1.0d99

lower bound for final flight azimuth (0 <= azimuth <= 360; degrees)
1.0d99

final declination (-90 <= declination <= +90; degrees)
+10.0

upper bound for final declination (-90 <= declination <= +90; degrees)
1.0d99

lower bound for final declination (-90 <= declination <= +90; degrees)
1.0d99

final east longitude (0 <= longitude <= 360; degrees)
30.0

upper bound for final east longitude (0 <= longitude <= 360; degrees)
1.0d99

lower bound for final east longitude (0 <= longitude <= 360; degrees)
1.0d99

The next series of data inputs define lower and upper bounds on the state variables during the aero-assist phase. To free or un-constrain one or more states, set the lower and/or upper bounds to 1.0d99.

upper and lower bounds on the flight conditions during phase

NOTE: set bound to 1.0d99 to ignore

upper bound for altitude (feet)
450000.0d0

```

lower bound for altitude (feet)
10000.0

upper bound for velocity (feet/second)
35000.0d0

lower bound for velocity (feet/second)
1000.0

upper bound for flight path angle (-90 <= fpa <= +90; degrees)
+20.0

lower bound for flight path angle (-90 <= fpa <= +90; degrees)
-20.0

upper bound for flight azimuth (0 <= azimuth <= 360; degrees)
1.0d99

lower bound for flight azimuth (0 <= azimuth <= 360; degrees)
1.0d99

upper bound for declination (-90 <= declination <= +90; degrees)
1.0d99

lower bound for declination (-90 <= declination <= +90; degrees)
1.0d99

upper bound for east longitude (0 <= longitude <= 360; degrees)
1.0d99

lower bound for east longitude (0 <= longitude <= 360; degrees)
1.0d99

```

This section of the input file defines the initial guesses and bounds for the control variables. To free or un-constrain one or more control variables, set the lower and/or upper bounds to 1.0d99.

```

*****
initial flight controls and bounds
*****
NOTE 1: set upper and lower bounds
to the initial value to constrain
or "fix" a flight control.
NOTE 2: set bound to 1.0d99 to ignore
-----

initial normalized lift coefficient
0.5

upper bound for initial normalized lift coefficient
+2.0

lower bound for initial normalized lift coefficient
+0.0d0

initial bank angle (degrees)
-90.0d0

upper bound for initial bank angle (degrees)
+0.0d0

```

lower bound for initial bank angle (degrees)
-180.0d0

This section of the input file defines the final guesses and bounds for the control variables. To free one or more control variables, set the lower and/or upper bounds to 1.0d99.

```
*****
final flight controls and bounds
*****
NOTE 1: set upper and lower bounds
to the final value to constrain
or "fix" a flight control.
NOTE 2: set bound to 1.0d99 to ignore
-----

final normalized lift coefficient
0.5

upper bound for final normalized lift coefficient
+2.0

lower bound for final normalized lift coefficient
+0.0

final bank angle (degrees)
-90.0d0

upper bound for final bank angle (degrees)
+0.0d0

lower bound for final bank angle (degrees)
-180.0d0
```

This section of the input file defines the bounds for the control variables during the aero-assist phase. To free or un-constrain one or more control variables, set the lower and/or upper bounds to 1.0d99.

```
*****
upper and lower bounds on the flight controls during phase
*****
NOTE: set bound to 1.0d99 to ignore
-----

upper bound for normalized lift coefficient
+2.0d0

lower bound for normalized lift coefficient
+0.0d0

upper bound for bank angle (degrees)
+0.0d0

lower bound for bank angle (degrees)
-180.0d0
```

This next section allows the user to define and enforce one or more point and path constraints during the atmospheric phase. Path constraints are enforced at all points along the trajectory and point constraints are enforced at atmospheric exit only. The user should be careful not to enforce constraints that are inconsistent with either the initial and/or final boundary conditions. For example, while maximizing the final orbital inclination do not enforce an orbital inclination point constraint.

```
*****
flight constraints
*****
```

enforce an orbital inclination constraint (yes or no)

yes

orbital inclination constraint value (degrees)

28.5d0

enforce a heating rate constraint (yes or no)

yes

heating rate upper bound constraint value (BTU/foot**2-second)

600.0d0

enforce a dynamic pressure constraint (yes or no)

no

dynamic pressure upper bound constraint value (pounds per square foot)

700.0d0

The type of trajectory initial guess or restart is specified by the next integer input. Program option 1 will use a simple linear initial guess created from the initial and final values provided by the user. Option 2 will read and use a binary file to initialize the simulation. Be sure to create a binary file first.

```
*****
initial guess/restart option
*****
 1 = linear guess
 2 = binary data file
-----
1
```

If the user elects to use a binary data file (option 2 above) for the initial guess, the following text input specifies the name of the file to use.

```
name of binary restart file
geo2leo_max_speed.rsbin
```

The following input can be used to create or update an initial guess binary file. The creation or update process uses the filename defined above. For initial guess options 1, the software will create a binary restart file. For initial guess option 2, an input of yes to this item will update the binary file used to initialize the simulation.

```
*****
binary restart file option
*****

create/update binary restart file (yes or no)
no
```

This next input specifies the type of comma-delimited or comma-separated-variable (CSV) solution data file to create. Option 1 will create a solution file at each collocation point or node determined by the AMA_OC software. Options 2 and 3 allow the user to specify either the number of nodes (option 2) or time step size of the data file (option 3).

```

*****
type of comma-delimited solution data file
*****
1 = OC-defined nodes
2 = user-defined nodes
3 = user-defined step size
-----
1

```

For options 2 or 3, this next input defines either the number of data points (option 2) or the time step size of the data output in the solution file (option 3).

```

number of user-defined nodes or print step size in solution data file
10.0

```

The software also creates a comma-separated-variable (csv) ASCII data file that contains the optimal control solution and many other flight parameters. The name of this output file is specified in the next line of information. Please consult Appendix A for additional information about the contents of this file.

```

name of solution output file
geo2leo_max_speed.csv

```

The next series of program inputs are algorithm control options and parameters for the AMA_OC software. The first input is an integer that specifies the type of collocation method to use during the solution process. For most simulations, the trapezoidal method is recommended.

```

*****
discretization/collocation method
*****
1 = trapezoidal
2 = separated Hermite-Simpson
3 = compressed Hermite-Simpson
-----
1

```

This input defines the relative error in the objective function.

```

relative error in the objective function (performance index)
1.0d-5

```

The next input defines the relative error in the solution of the differential equations.

```

relative error in the solution of the differential equations
1.0d-7

```

The next input is an integer that defines the maximum number of mesh refinement iterations.

```

maximum number of mesh refinement iterations
20

```

The next input is an integer that defines the maximum number of function evaluations.

```

maximum number of function evaluations
100000

```

The next input is an integer that defines the maximum number of algorithm iterations.

```

maximum number of algorithm iterations
10000

```

The level of output from the NLP algorithm is controlled with the following integer input.

```

*****
sparse NLP iteration output
*****
1 = none
2 = terse
3 = standard
4 = interpretive
5 = diagnostic
-----
2

```

The level of output from the optimal control algorithm is controlled with the following integer input. Please note that option 4 will create lots of information.

```

*****
optimal control output
*****
1 = none
2 = terse
3 = standard
4 = interpretive
-----
1

```

The level of output from the differential equations algorithm is controlled with the following integer input. Please note that option 5 will create lots of information.

```

*****
differential equation output
*****
1 = none
2 = terse
3 = standard
4 = interpretive
5 = diagnostic
-----
1

```

The level of output can be further controlled by the user with this final text input. This program option sets the value of the SOCOUT character variable described in the AMA_OC software user's manual. To ignore this special output control, input the simple character string no.

```

*****
user-defined output
-----
input no to ignore
*****
a0b0c0d0e0f0g0h0i0j1k0l0m0n0o0p0q0r0

```

The last series of inputs allow the reading and writing of configuration input files. The user should create a configuration file before attempting to read one. These configuration files are simple text files which can be edited external to the aeroassist software. Please consult Appendix F.

```

*****
* optimal control configuration options
*****

read an optimal control configuration file (yes or no)
no

```

```
name of optimal control configuration file
aeroassist_config1.txt

create an optimal control configuration file (yes or no)
no

name of optimal control configuration file
aeroassist_config1.txt
```

Atmosphere model and constants data file

The `aeroassist` software also requires a user-created ASCII data file with the name `tutility.dat` that defines the fundamental constants that will be used during the simulation. The following is the companion data file for this example. If the user provides a value of zero for the Earth's rotation rate, the simulation results will be with respect to a non-rotating, spherical Earth.

```
*****
simulation environment and
planet/utility constants
*****

radius of the earth (feet)
20925656.8d0

gravitational constant of the earth (feet**3/second**2)
1.407644381252d16

surface gravity (feet/second**2)
32.174d0

earth rotation rate (radians/second)
7.2921151467d-5

atmospheric surface density (slugs/feet**3)
0.0023765d0

j2 gravity coefficient (non-dimensional)
1.08262668355d-3

j3 gravity coefficient (non-dimensional)
0.0d0

j4 gravity coefficient (non-dimensional)
0.0d0
```

The user's input for `j2`, `j3` and `j4` control the type of gravity model used during the simulation.

Optimal control solution and graphics

After the `aeroassist` scientific simulation has converged, it will display a complete summary of the initial conditions and the optimized trajectory. It also provides a summary of the relative flight path coordinates and the aerodynamic characteristics of the vehicle. The classical orbital elements at atmospheric exit are determined from the inertial state vector which is computed using the relative flight path coordinates at exit.

The following is a summary of the solution for this example.

program aeroassist
=====

input file ==> geo2leo_max_speed.in

bounded entry conditions derived from deorbit maneuver

maximize speed at atmospheric exit

orbital elements and state vector prior to deorbit impulse

calendar date	January	1,	2001				
universal time	00:00:00.000						
sma (nm)	eccentricity	inclination (deg)	argper (deg)				
0.227669201902D+05	0.111022302463D-15	0.000000000000D+00	0.000000000000D+00				
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)				
0.000000000000D+00	0.300000000000D+02	0.300000000000D+02	0.143607658003D+04				
r-perigee (nm)	h-perigee (nm)	r-apogee (nm)	h-apogee (nm)				
0.227669201902D+05	0.193230000000D+05	0.227669201902D+05	0.193230000000D+05				
rx (ft)	ry (ft)	rz (ft)	rmag (ft)				
0.119801136085D+09	0.691672181680D+08	0.000000000000D+00	0.138334436336D+09				
vx (fps)	vy (fps)	vz (fps)	vmag (fps)				
-.504372416457D+04	0.873598651240D+04	0.000000000000D+00	0.100874483291D+05				

orbital elements and state vector after deorbit impulse

calendar date	January	1,	2001				
universal time	00:00:00.000						
sma (nm)	eccentricity	inclination (deg)	argper (deg)				
0.131213107012D+05	0.735110211827D+00	0.000000000000D+00	0.210000000000D+03				
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)				
0.000000000000D+00	0.180000000000D+03	0.300000000000D+02	0.628328849946D+03				
r-perigee (nm)	h-perigee (nm)	r-apogee (nm)	h-apogee (nm)				
0.347570121219D+04	0.317810220239D+02	0.227669201902D+05	0.193230000000D+05				
rx (ft)	ry (ft)	rz (ft)	rmag (ft)				
0.119801136085D+09	0.691672181680D+08	0.000000000000D+00	0.138334436336D+09				
vx (fps)	vy (fps)	vz (fps)	vmag (fps)				
-.259587595393D+04	0.449618904236D+04	0.000000000000D+00	0.519175190787D+04				

flight path coordinates at atmospheric entry

altitude	400000.000000000	feet
velocity	32268.5194754447	feet/second
declination	2.637383834618052E-013	degrees
longitude	18.8080427045337	degrees
azimuth	90.0000000000005	degrees
flight path angle	-5.45162080704044	degrees

inertial fpa -5.20130119814137 degrees

orbital elements and state vector at atmospheric entry

calendar date January 1, 2001

universal time 05:11:55.842

sma (nm)	eccentricity	inclination (deg)	argper (deg)
0.131213104261D+05	0.735110206972D+00	0.581675704578D-12	0.210000003999D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.000000000000D+00	0.347714838486D+03	0.197714842485D+03	0.628328830184D+03
r-perigee (nm)	h-perigee (nm)	r-apogee (nm)	h-apogee (nm)
0.347570120302D+04	0.317810128543D+02	0.227669196491D+05	0.193229994589D+05
rx (ft)	ry (ft)	rz (ft)	rmag (ft)
-.203144515089D+08	-.648896739412D+07	0.981641981048D-07	0.213256568000D+08
vx (fps)	vy (fps)	vz (fps)	vmag (fps)
0.131677427163D+05	-.311479253279D+05	-.318848277882D-09	0.338168996284D+05

flight path coordinates at atmospheric exit

altitude	400000.000000000	feet
velocity	25600.1592225824	feet/second
declination	11.8749410000429	degrees
longitude	59.7589681113454	degrees
azimuth	62.4006932817757	degrees
flight path angle	2.85290494398532	degrees

orbital elements and state vector at atmospheric exit

calendar date January 1, 2001

universal time 05:21:20.255

sma (nm)	eccentricity	inclination (deg)	argper (deg)
0.390347950906D+04	0.111289864741D+00	0.285000000002D+02	0.357704478388D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.218238520146D+03	0.278426340657D+02	0.255471124534D+02	0.101952769539D+03
r-perigee (nm)	h-perigee (nm)	r-apogee (nm)	h-apogee (nm)
0.346906180248D+04	0.251416123121D+02	0.433789721564D+04	0.893977025480D+03
rx (ft)	ry (ft)	rz (ft)	rmag (ft)
-.101099984601D+08	-.182568968034D+08	0.438831268249D+07	0.213256568000D+08
vx (fps)	vy (fps)	vz (fps)	vmag (fps)
0.217306749776D+05	-.106726700692D+05	0.118541683058D+05	0.269564357364D+05

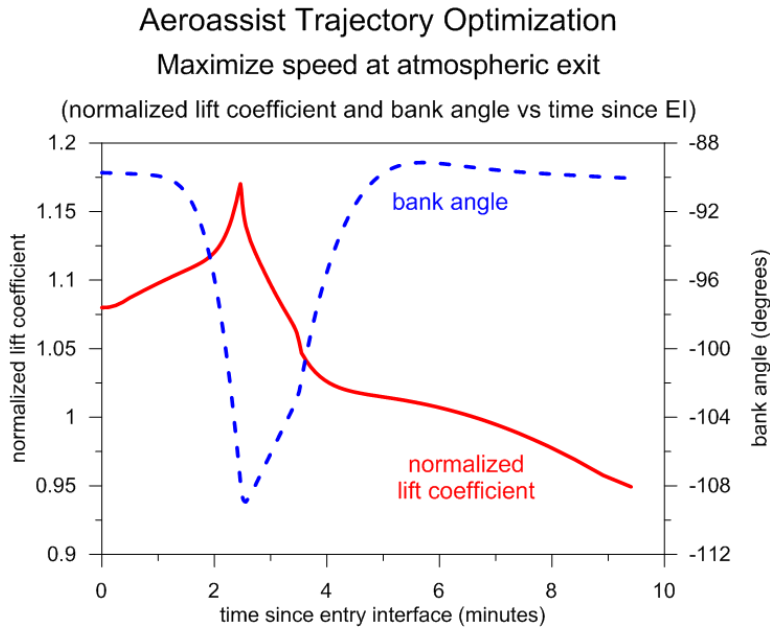
aerodynamic characteristics

drag coefficient at aoa = 0 degrees 3.200000000000000E-002

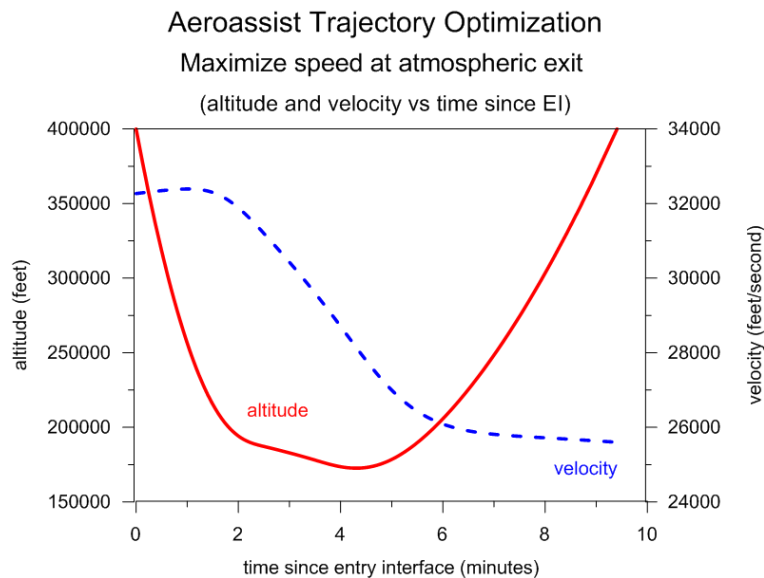
drag coefficient at max L/D 6.400000000000000E-002

lift coefficient at max L/D 0.151185789203691
maximum lift-to-drag ratio 2.36227795630767

The following are plots created from the trajectory summary file. The first plot illustrates the behavior of the normalized lift coefficient and bank angle during the atmospheric pass.



This next plot summarizes the altitude and relative velocity of the vehicle as a function of time since the entry interface (EI).

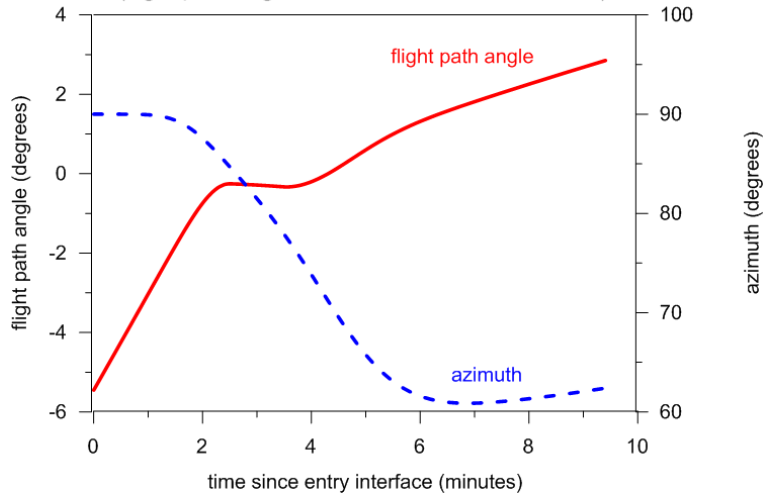


The next plot illustrates the behavior of the relative flight path and azimuth angles during the aero-assist pass through the atmosphere.

Aeroassist Trajectory Optimization

Maximize speed at atmospheric exit

(flight path angle and azimuth vs time since EI)

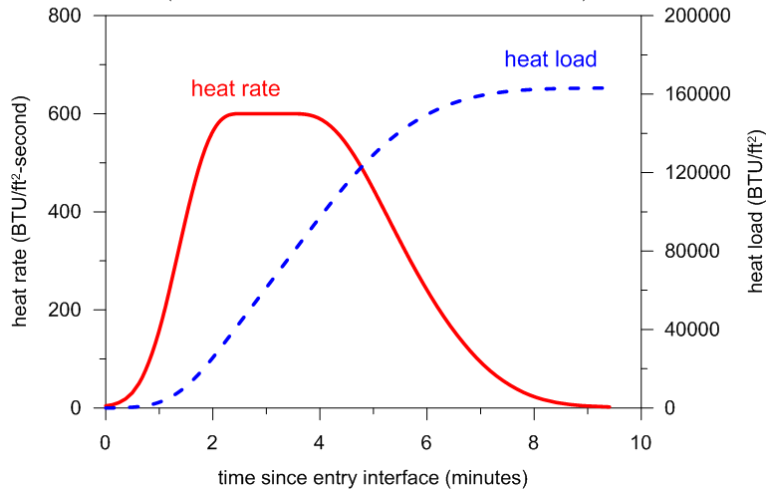


This plot illustrates the behavior of the heat rate and heat load during the atmospheric portion of the trajectory. It confirms that the solution has satisfied the maximum heat rate path constraint.

Aeroassist Trajectory Optimization

Maximize speed at atmospheric exit

(heat rate and heat load vs time since EI)

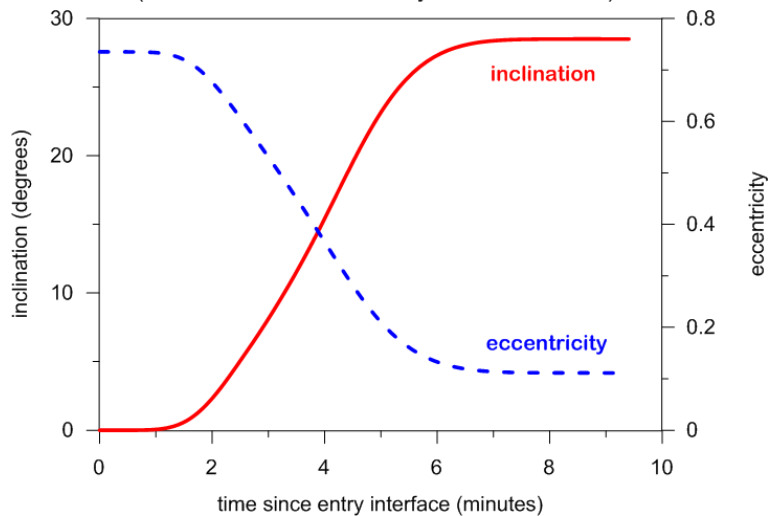


This final plot summarizes the behavior of the orbital eccentricity and inclination during the atmospheric phase of the mission.

Aeroassist Trajectory Optimization

Maximize speed at atmospheric exit

(inclination and eccentricity vs time since EI)



Problem setup

This section provides additional details about the *AMA_OC* software implementation. It briefly explains such things as initial conditions, path constraints and the performance index options.

(1) Entry interface initial conditions

The *aeroassist* computer program includes three options for specifying initial conditions at the entry interface (EI). This section summarizes these options and describes how the user invokes each.

(a) user input of flight path coordinates at entry interface

For this option, all Earth relative coordinates at the entry interface are defined by the user. The user can provide initial guesses and lower and upper bounds for these coordinates in the `flight conditions` and `bounds at atmospheric entry` part of the input data file.

(b) derived from deorbit maneuver; fixed entry conditions

For this program option, the software uses the flight path angle and entry altitude provided in the input data file to constrain the entry altitude and *inertial* flight path angle. The software then uses the deorbit algorithm described later in this section to compute the Earth relative flight path angle, speed and other flight path coordinates at the entry interface. This option is valid for initial circular orbits.

(c) derived from deorbit maneuver; bounded entry conditions

For this program option, the software will use the entry altitude provided in the input data file to constrain the entry altitude. The flight path angle provided in the data file is used for an *inertial* flight path angle initial guess. During the trajectory optimization, the software will change the inertial flight path angle between the lower and upper bounds provided by the user (the inertial flight path angle is

treated as a problem parameter). The software then uses the deorbit algorithm described later in this section to compute the Earth relative flight path angle, speed and other coordinates at the entry interface. This option is valid for initial circular orbits.

For the second and third initial conditions options, the `aeroassist` computer program calculates the single impulsive maneuver required to establish an entry interface altitude and flight path angle relative to the user-defined initial circular orbit. The algorithm uses a tangential ΔV applied opposite to the velocity vector to establish the deorbit trajectory. The entry altitude and flight path angle initial guesses are provided by the user.

The algorithm used to compute the scalar magnitude of the deorbit maneuver along with other important flight characteristics is described in Appendix E.

(2) Performance index

This section describes the two types of trajectory optimization performed by the `aeroassist` software.

(a) *maximize final speed*

The performance index for this type of optimization is simply

$$J = v_f$$

where v_f is the relative speed of the vehicle as it exits from the atmosphere. For this program option, the optimization indicator is set to `maxmin = +1`. This option minimizes the energy loss during the aero-assist maneuver.

(b) *maximize inclination change*

The performance index for this program option is given by

$$J = \cos^{-1}\left(\frac{h_z}{h}\right)$$

where h_z is the z-component of the ECI angular momentum vector and h is the angular momentum magnitude of the vehicle at exit from the atmosphere at the user-defined altitude. For this program option, the optimization indicator is also set to `maxmin = +1`.

(3) Path and point constraints

This section summarizes how the software computes the heat rate and dynamic pressure path constraints, and the orbital inclination point constraint.

(a) *Dynamic pressure*

To enforce this path constraint, the software ensures that

$$q \leq q_{\max}$$

where q_{\max} is the user-defined value of the maximum dynamic pressure. The dynamic pressure at any simulation time is given by

$$q = \frac{1}{2} \rho v^2$$

where ρ is the atmospheric density and v is the relative speed at the current flight condition.

(b) *Heat rate*

To enforce this path constraint, the software ensures that

$$\dot{Q} \leq \dot{Q}_{\max}$$

where \dot{Q}_{\max} is the user-defined value of the maximum heat rate. The heat rate at any simulation time is computed from Chapman's stagnation point equation given by

$$\dot{Q} = \frac{dQ}{dt} = \frac{17,600}{\sqrt{R_N}} \left(\frac{V}{V_0} \right)^{3.15} \sqrt{\frac{\rho}{\rho_0}} \quad \left(\frac{\text{BTU}}{\text{ft}^2 - \text{sec}} \right)$$

where

R_N = nose radius (feet)

V = relative velocity at the spacecraft location (feet/second)

V_0 = "local circular velocity" at the Earth's surface = $\sqrt{\mu/r_e}$ (feet/second)

ρ = atmospheric density at the spacecraft location (slugs/feet³)

ρ_0 = atmospheric density at the Earth's surface (slugs/feet³)

μ = gravitational constant of the Earth (feet³/second²)

r_e = radius of the Earth (feet)

(c) *Orbital inclination*

A final orbital inclination point constraint is enforced as follows

$$\cos i = \frac{h_z}{h}$$

where h_z is the z-component of the angular momentum vector and h is the angular momentum magnitude at the atmospheric exit. In this equation, i is the user-defined final orbit inclination.

Technical Discussion

This section describes the numerical algorithms implemented in the `aeroassist` computer program. It summarizes the equations of motion, Earth gravity model, vehicle aerodynamics, coordinate conversions and other important software features.

Flight path equations of motion

The first-order flight path equations of motion of an aerospace vehicle relative to a rotating spherical Earth and a zonal gravity model are summarized as follows:

geocentric radius

$$\dot{r} = \frac{dr}{dt} = V \sin \gamma$$

geographic longitude

$$\dot{\lambda} = \frac{d\lambda}{dt} = V \frac{\cos \gamma \sin \psi}{r \cos \delta}$$

geocentric declination

$$\dot{\delta} = \frac{d\delta}{dt} = V \frac{\cos \gamma \cos \psi}{r}$$

speed

$$\begin{aligned} \dot{V} = \frac{dV}{dt} = & \frac{(T \cos \alpha - D)}{m} - g_r \sin \gamma + g_\phi \cos \gamma \cos \psi \\ & + \omega_e^2 r \cos \delta (\sin \gamma \cos \delta - \sin \delta \cos \gamma \cos \psi) \end{aligned}$$

flight path angle

$$\begin{aligned} \dot{\gamma} = \frac{d\gamma}{dt} = & \frac{V}{r} \cos \gamma + \left(\frac{T \sin \alpha + L}{mV} \right) \cos \beta - \frac{g_r \cos \gamma}{V} - \frac{g_\phi \sin \gamma \cos \psi}{V} \\ & + 2\omega_e \sin \psi \cos \delta + \omega_e^2 \frac{r}{V} \cos \delta (\cos \psi \sin \gamma \sin \delta + \cos \gamma \cos \delta) \end{aligned}$$

flight azimuth

$$\begin{aligned} \dot{\psi} = \frac{d\psi}{dt} = & \frac{V}{r} \tan \delta \sin \psi \cos \gamma + \left(\frac{T \sin \alpha + L}{mV \cos \gamma} \right) \sin \beta - \frac{\sin \psi}{V \cos \gamma} g_\phi \\ & + 2\omega_e (\sin \delta - \cos \psi \cos \delta \tan \gamma) + \frac{r}{V \cos \gamma} \omega_e^2 \sin \psi \cos \delta \sin \delta \end{aligned}$$

where

- r = geocentric radius
- V = speed
- γ = flight path angle
- δ = geocentric declination
- λ = longitude (+ east)
- ψ = flight azimuth (+ clockwise from north)
- β = bank angle (+ for a right turn)
- α = angle of attack
- L = aerodynamic lift force = $\frac{1}{2}\rho V^2 C_L S$
- D = aerodynamic drag force = $\frac{1}{2}\rho V^2 C_D S$
- T = propulsive thrust
- m = spacecraft mass
- C_L = lift coefficient (non-dimensional)
- C_D = drag coefficient (non-dimensional)
- S = aerodynamic reference area
- ρ = atmospheric density = $f(h)$
- h = altitude

and

- r_e = Earth equatorial radius
- ω_e = Earth inertial rotation rate
- g_r = radial component of gravity
- g_ϕ = latitudinal component of gravity

The bank angle is the angle between the instantaneous orbit plane and the aerodynamic lift vector. The bank angle is positive for a right turn as viewed from the rear of the vehicle.

Earth gravity model

The components of the gravity vector can be determined from the gradient of the potential function according to

$$\mathbf{F}_G = \nabla U = \begin{Bmatrix} \frac{1}{r} \frac{\partial U}{\partial \phi} \\ 0 \\ -\frac{\partial U}{\partial r} \end{Bmatrix} = \begin{Bmatrix} g_\phi \\ 0 \\ g_r \end{Bmatrix}$$

where

$$U = \frac{\mu}{r} \left[1 - \sum_{j=1}^{\infty} \left(\frac{r_e}{r} \right)^j J_j P_{j0}(\sin \phi) \right]$$

$$\frac{\partial U}{\partial r} = \frac{\mu}{r^2} \left[-1 + \sum_{l=2}^{\infty} (l+1) \left(\frac{r_e}{r} \right)^l J_l P_{l0}(\sin \phi) \right]$$

$$\frac{1}{r} \frac{\partial U}{\partial \phi} = -\frac{\mu}{r^2} \sum_{l=2}^{\infty} \left(\frac{r_e}{r} \right)^l J_l \frac{\partial P_{l0}}{\partial \phi}$$

$$P_{j0}(\sin \phi) = \sum_{t=0}^k T_{jt} \sin^{j-2t} \phi$$

$$T_{jt} = (-1)^t (2j-2t)! / 2t! (j-t)! (j-2t)!$$

For a *zonal-only* gravity model of order four, the Legendre functions and their partial derivatives are given by

$$P_{20} = \frac{1}{2} (3 \sin^2 \phi - 1) \quad P_{30} = \frac{1}{2} (5 \sin^3 \phi - 3 \sin \phi) \quad P_{40} = \frac{1}{8} (35 \sin^4 \phi - 30 \sin^2 \phi + 3)$$

$$\frac{\partial P_{20}}{\partial \phi} = 3 \sin \phi \cos \phi \quad \frac{\partial P_{30}}{\partial \phi} = \frac{3}{2} (5 \sin^2 \phi - 1) \cos \phi \quad \frac{\partial P_{40}}{\partial \phi} = \frac{5}{2} (7 \sin^2 \phi - 3) \sin \phi \cos \phi$$

This is the Earth gravity model implemented in the `aeroassist` computer program.

Coordinate systems and transformations

This section describes the relationship between flight path coordinates and inertial position and velocity. Flight path coordinates are used during the aero-assist pass and inertial coordinates are used to model the deorbit-to-entry portion of the flight.

(1) converting an ECI state vector to flight path coordinates

This coordinate conversion is necessary for `aeroassist` simulations that determine entry interface conditions from an impulsive deorbit maneuver from an initial circular orbit.

The transformation of an ECI position vector \mathbf{r}_{eci} to an ECF position vector \mathbf{r}_{ecf} is given by the following vector-matrix operation

$$\mathbf{r}_{ECF} = [\mathbf{T}] \mathbf{r}_{ECI}$$

where the elements of the transformation matrix $[\mathbf{T}]$ are given by

$$[\mathbf{T}] = \begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

and θ is the Greenwich apparent sidereal time at the moment of interest. Greenwich apparent sidereal time is given by the following expression:

$$\theta = \theta_{g_0} + \omega_e t$$

where θ_{g_0} is the Greenwich apparent sidereal time at 0 hours UT, ω_e is the inertial rotation rate of the Earth, and t is the elapsed time since 0 hours UT.

Finally, the flight path coordinates are determined from the following set of equations

$$\begin{aligned} r &= \sqrt{r_{ECF_x}^2 + r_{ECF_y}^2 + r_{ECF_z}^2} & v &= \sqrt{v_{ECF_x}^2 + v_{ECF_y}^2 + v_{ECF_z}^2} \\ \lambda &= \tan^{-1}(r_{ECF_y}, r_{ECF_x}) & \delta &= \sin^{-1}\left(\frac{r_{ECF_z}}{|\mathbf{r}_{ECF}|}\right) \\ \gamma &= \sin^{-1}\left(-\frac{v_{r_z}}{|\mathbf{v}_r|}\right) & \psi &= \tan^{-1}[v_{r_y}, v_{r_x}] \end{aligned}$$

where

$$\mathbf{v}_r = \begin{bmatrix} -\sin \delta \cos \lambda & -\sin \delta \sin \lambda & \cos \delta \\ -\sin \lambda & \cos \lambda & 0 \\ -\cos \delta \cos \lambda & -\cos \delta \sin \lambda & -\sin \delta \end{bmatrix} \mathbf{v}_{ECF}$$

(2) converting flight path coordinates to an ECI state vector

This coordinate conversion is necessary in order to determine the orbital inclination and other orbital characteristics at exit from the atmosphere.

The Earth-centered-fixed (ECF) position and velocity vectors of the aero-assist vehicle can be determined from the flight path coordinates with the following set of equations:

$$r_{ECF_x} = r \cos \delta \cos \lambda$$

$$r_{ECF_y} = r \cos \delta \sin \lambda$$

$$r_{ECF_z} = r \sin \delta$$

$$v_{ECF_x} = v \left[\cos \lambda (-\cos \psi \sin \beta \sin \delta + \cos \beta \cos \delta) - \sin \psi \sin \beta \sin \lambda \right]$$

$$v_{ECF_y} = v \left[\sin \lambda (-\cos \psi \sin \beta \sin \delta + \cos \beta \cos \delta) + \sin \psi \sin \beta \cos \lambda \right]$$

$$v_{ECF_z} = v (\cos \psi \cos \delta \sin \beta + \cos \beta \cos \delta)$$

where $\beta = 90^\circ - \gamma$.

The transformation from ECF to ECI coordinates involves the transpose of the ECI-to-ECF transformation matrices described above as follows:

$$\mathbf{r}_{ECI} = [\mathbf{T}]^T \mathbf{r}_{ECF}$$

$$\mathbf{v}_{ECI} = [\mathbf{T}]^T \dot{\mathbf{r}}_{ECF} + [\dot{\mathbf{T}}]^T \mathbf{r}_{ECF} = [\mathbf{T}]^T \mathbf{v}_{ECF} + [\dot{\mathbf{T}}]^T \mathbf{r}_{ECF}$$

The ECF velocity vector is determined by differentiating this expression

$$\mathbf{v}_{ECF} = [\mathbf{T}] \dot{\mathbf{r}}_{ECI} + [\dot{\mathbf{T}}] \mathbf{r}_{ECI} = [\mathbf{T}] \mathbf{v}_{ECI} + [\dot{\mathbf{T}}] \mathbf{r}_{ECI}$$

The elements of the $[\dot{\mathbf{T}}]$ matrix are as follows:

$$[\dot{\mathbf{T}}] = \begin{bmatrix} -\omega_e \sin \theta & \omega_e \cos \theta & 0 \\ -\omega_e \cos \theta & -\omega_e \sin \theta & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Aerodynamic characteristics of aero-assist vehicles

This section provides general information about vehicle aerodynamics. It also describes the type of aerodynamic modeling used in the `aeroassist` computer program.

General form of a drag polar

$$C_D = C_{D_0} + k|C_L|^n$$

Lift-to-drag ratio

$$E = \frac{L}{D} = \frac{C_L}{C_D} = \frac{C_L}{C_{D_0} + kC_L^n}$$

Maximum lift-to-drag ratio (value of E at which $dE/dC_L = 0$)

$$E^* = \frac{(C_{D_0} + kC_L^n) - C_L(nkC_L^{n-1})}{(C_{D_0} + kC_L^n)^2}$$

Lift coefficient at maximum lift-to-drag ratio

$$C_L^* = \sqrt[n]{\frac{C_{D_0}}{k(n-1)}}$$

Drag coefficient at maximum lift-to-drag ratio

$$C_D^* = \frac{nC_{D_0}}{(n-1)}$$

In general,

$$E^* = \frac{C_L^*}{C_D^*} = \frac{\sqrt[n]{(n-1)^{n-1}}}{n\sqrt[n]{kC_{D_0}^{n-1}}}$$

For a *parabolic* drag polar ($n = 2$),

$$C_D = C_{D_0} + kC_L^2$$

where

C_D = drag coefficient

C_{D_0} = drag coefficient at angle-of-attack = 0°

C_L = lift coefficient

k = constant

and

$C_D^* = 2C_{D_0}$ = drag coefficient at maximum L/D

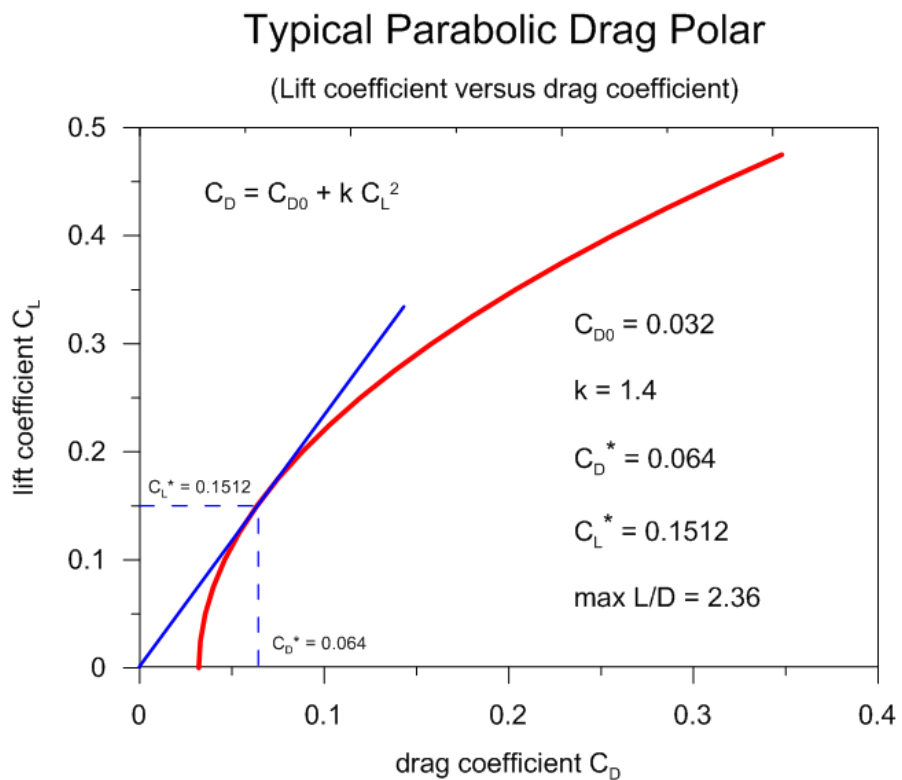
$C_L^* = \sqrt{C_{D_0}/k}$ = lift coefficient at maximum L/D

$E^* = \left(\frac{C_L}{C_D}\right)_{\max} = \frac{1}{2\sqrt{kC_{D_0}}} = \text{maximum L/D}$

In the *aeroassist* computer program, the vehicle aerodynamics are modeled using a parabolic drag polar ($n = 2$). The normalized lift coefficient is given by $\lambda = C_L/C_L^*$ where C_L is the lift coefficient at any simulation time and C_L^* is the lift coefficient corresponding to maximum L/D.

In this computer program, the control variables are the normalized lift coefficient and bank angle.

The following is a graphic display of a typical parabolic drag polar.



References and Bibliography

- (1) "Direct Trajectory Optimization Using Nonlinear Programming and Collocation", C. R. Hargraves and S. W. Paris, *AIAA Journal of Guidance, Control and Dynamics*, Vol. 10, No. 4, July-August, 1987, pp. 338-342.
- (2) "Optimal Finite-Thrust Spacecraft Trajectories Using Direct Transcription and Nonlinear Programming", Paul J. Enright, Ph.D. Thesis, University of Illinois at Urbana-Champaign, 1991.
- (3) "Using Sparse Nonlinear Programming to Compute Low Thrust Orbit Transfers", John T. Betts, *The Journal of the Astronautical Sciences*, Vol. 41, No. 3, July-September 1993, pp. 349-371.
- (5) "Improved Collocation Methods with Application to Direct Trajectory Optimization", Albert L. Herman, Ph.D. Thesis, University of Illinois at Urbana-Champaign, 1995.
- (6) "Minimum-Fuel Aero-assisted Coplanar Orbit Transfer Using Lift Modulation", Kenneth D. Mease and Nguyen X. Vinh, *AIAA Journal of Guidance, Control and Dynamics*, Vol. 8, No. 1, Jan.-Feb. 1985.
- (7) "Survey of Numerical Methods for Trajectory Optimization", John T. Betts, *AIAA Journal of Guidance, Control and Dynamics*, Vol. 21, No. 2, March-April 1998, pp. 193-207.
- (8) "Variational Solutions for the Heat-Rate-Limited Aero-assisted Orbital Transfer Problem", Hans Seywald, *AIAA Journal of Guidance, Control and Dynamics*, Vol. 19, No. 3, May-June, 1996, pp. 686-692.
- (9) "Fuel-Optimal Trajectories of Aero-assisted Orbital Transfer with Plane Change", D. S. Naidu, *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 27, No. 2, March 1991, pp. 361-368.
- (10) *Optimal Trajectories in Atmospheric Flight*, N. X. Vinh, Elsevier, 1981.
- (11) *Hypersonic and Planetary Flight Mechanics*, N. X. Vinh, R. D. Culp, and A. Busemann, University of Michigan Press, 1980.
- (12) "A Survey of Aero-assisted Orbit Transfer", G. D. Walberg, *AIAA Journal of Spacecraft and Rockets*, Vol. 22, No. 1, Jan-Feb 1985, pp. 3-18.
- (13) "Optimal Maneuvers of Orbital Transfer Vehicles", John M. Hanson, Ph.D. Thesis, University of Michigan, Department of Aerospace Engineering, 1983.
- (14) "Optimal Aero-assisted Return From High Earth Orbit With Plane Change", Nguyen X. Vinh and John M. Hanson, *Acta Astronautica*, Vol. 12, No. 1, 1985.
- (15) "Combining Propulsive and Aerodynamic Maneuvers to Achieve Optimal Orbital Transfer", John M. Hanson, *AIAA Journal of Guidance, Control and Dynamics*, Vol. 12, No. 5, Sept.-Oct. 1989.

APPENDIX A

Contents of the Simulation Summary CSV File

This appendix is a brief summary of the information contained in the CSV data file produced by the `aeroassist` software. The comma-separated-variable disk file is created by the `odeprt` subroutine and contains the following information:

time (min) = simulation time since entry interface in minutes
altitude (ft) = altitude relative to a spherical Earth in feet
velocity (fps) = Earth-relative velocity in feet per second
flight path angle (d) = Earth-relative flight path angle in degrees
azimuth (deg) = Earth-relative azimuth angle in degrees
declination (deg) = geocentric declination in degrees
longitude (deg) = geographic longitude in degrees
mach number = Mach number (non-dimensional)
dynamic pressure (psf) = dynamic pressure in pounds per square foot
bank angle (deg) = bank angle in degrees
heat rate (btu/ft²-s) = heat rate in BTU/ft²-second
heat load (btu/ft²) = accumulated heat load in BTU/ft²
lift-to-drag = lift-to-drag ratio (non-dimensional)
lift coefficient = lift coefficient (non-dimensional)
drag coefficient = drag coefficient (non-dimensional)
lift force (lbf) = lift force in pounds
drag force (lbf) = drag force in pounds
density (slugs/ft³) = atmospheric density in slugs per cubic feet
pressure (psf) = atmospheric pressure in pounds per square foot
temperature (deg K) = atmospheric temperature in degrees K
crossrange (nm) = crossrange distance in nautical miles
downrange (nm) = downrange distance in nautical miles
altitude rate (fps) = rate of change of altitude in feet per second
longitude rate (dps) = rate of change of longitude in degrees per second
declination rate (dps) = rate of change of declination in degrees per second
velocity rate (fps/s) = rate of change of velocity in feet per second per second
fpa rate (dps) = rate of change of flight path angle in degrees per second
azimuth rate (dps) = rate of change of azimuth angle in degrees per second

perigee altitude (nm) = perigee altitude in nautical miles

perigee radius (nm) = perigee radius in nautical miles

apogee altitude (nm) = apogee altitude in nautical miles

apogee radius (nm) = apogee radius in nautical miles

Notes:

- (1) The accumulated heat load is determined from a cubic spline integration of the heat rate of the optimized solution at all collocation nodes.
- (2) The rate of change of the flight variables is determined from the equations of motion.

APPENDIX B

Fortran Functions and Subroutines

This appendix is a brief summary of the major Fortran functions and subroutines included in the `aeroassist` computer program.

`aeroassist.f` - main executive program

`atan3.for` - four quadrant inverse tangent function

`atmos76.for` - U.S. Standard 1976 atmosphere model

`cdeorbit.for` - impulsive deorbit from a circular orbit subroutine

`crdr.for` - subroutine that calculates crossrange and downrange

`csint.for` - cubic spline integration of tabular data subroutine

`gast.for` - Greenwich apparent sidereal time subroutine

`gravity.for` - fourth-order zonal gravity model subroutine

`odeinp.for` - simulation input subroutine

`odepf.for` - point functions subroutine

`odeprt.for` - print subroutine - creates comma-separated-variable file

`oderhs.for` - subroutine that evaluates the equations of motion and any algebraic equations

`readfpn.for` - read and echo floating point number from an input file subroutine

`readint.for` - read and echo an integer from an input file subroutine

`readtext.for` - read and echo text from an input file subroutine

`twobody2.for` - two-body orbit propagation subroutine

`utility.for` - number and text manipulation functions and subroutines

`us76.for` - U.S. standard 1976 atmosphere subroutine

`uvector.for` - unit vector subroutine

`vcross.for` - vector cross product subroutine

`vdot.for` - vector dot product subroutine

`vecmag.for` - vector scalar magnitude function

`xmod.for` - modulo 2 pi function

APPENDIX C

Example Fortran Subroutine

This appendix contains the source code for a single Fortran 77 subroutine and illustrates typical programming conventions used in the `aeroassist` software. This subroutine is the point function routine required by the `AMA_OC` software.

```
      subroutine odepf(iphase, iphend, time, ydyn, nydyn, parm,
&                   nparm, ptf, nptf, iferr)
c
c   aeroassist point functions
c   *****
      implicit double precision (a-h, o-z)
      include 'socscom1.inc'
      include 'pconstr.inc'
      parameter (zero = 0.0d0, one = 1.0d0)
      dimension ydyn(nydyn), parm(nparm), ptf(nptf), ywrk(6)
      dimension reci(3), veci(3), fpc(6), hv(3), oev(6)
      iferr = 0
c
c   -----
c   extract current flight path coordinates
c   -----
c
c   altitude (feet)
      xalt = ydyn(1)
c
c   longitude (radians)
      elon = ydyn(2)
c
c   geocentric declination
      dec = ydyn(3)
c
c   relative speed (feet/second)
      vrel = ydyn(4)
c
c   flight path angle (radians)
      fpa = ydyn(5)
c
c   flight azimuth (radians)
      azim = ydyn(6)
c
c   geocentric radius (feet)
      rmag = xalt + req
      if (iphase .eq. 1 .and. iphend .eq. -1
&       .and. ic_type .eq. 3) then
```

```

c -----
c beginning of phase 1 - atmospheric entry
c (parameter #1 ==> inertial flight path angle)
c -----

c current inertial flight path angle

fpae = parm(1)

c compute flight path coordinates at entry interface
c using user-defined entry altitude and orbital elements

call cdeorbit(fpae, ywrk)

c -----
c match states at atmospheric entry
c -----

c east longitude (radians)

ptf(1) = ydyn(2) - ywrk(2)

c declination (radians)

ptf(2) = ydyn(3) - ywrk(3)

c velocity (feet/second)

ptf(3) = ydyn(4) - ywrk(4)

c flight path angle (radians)

ptf(4) = ydyn(5) - ywrk(5)

c azimuth (radians)

ptf(5) = ydyn(6) - ywrk(6)

end if

if (iphase .eq. 1 .and. iphend .eq. +1) then

c -----
c end of phase 1 - atmospheric exit
c -----

c compute inertial state vector at end of phase 1

fpc(1) = elon
fpc(2) = dec
fpc(3) = fpa
fpc(4) = azim
fpc(5) = rmag
fpc(6) = vrel

call fpc2eci(time, fpc, reci, veci)

c compute angular momentum vector and magnitude

call vcross(reci, veci, hv)

```

```

hmag = vecmag(hv)

if (ic_inc .eq. 1) then

c      -----
c      final orbit inclination constraint at atmospheric exit
c      (constrain cosine of final orbit inclination)
c      -----

      ptf(1) = hv(3) / hmag

end if

if (iopt .eq. 2) then

c      -----
c      maximize orbital inclination at atmospheric exit
c      -----

      ptf(1) = acos(hv(3) / hmag)

end if

end if

return
end

```

APPENDIX D

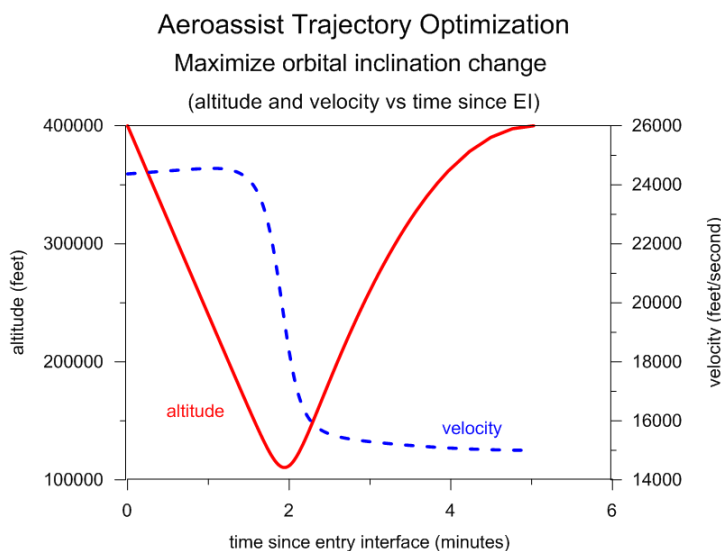
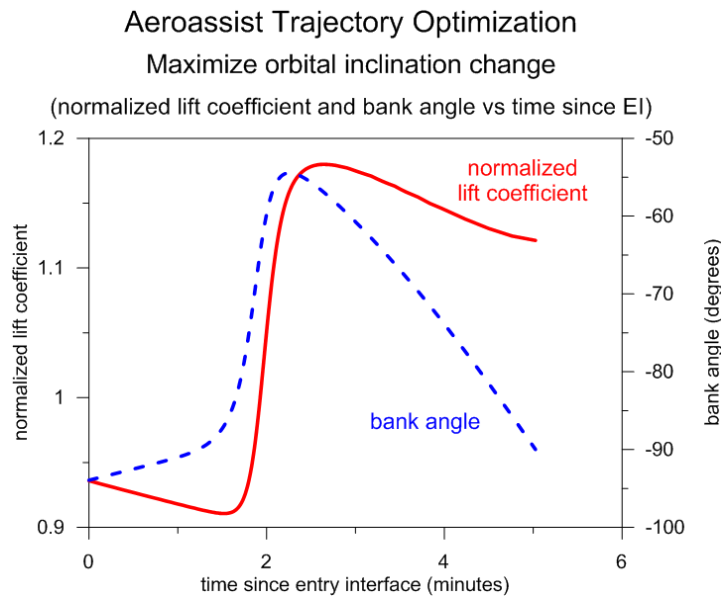
Maximize Orbital Inclination Example

This appendix contains graphics and a simulation summary for an aero-assist trajectory that maximizes the orbital inclination change. The mission starts in a 500 nautical mile circular Earth orbit with an initial inclination equal to 5 degrees. The speed at atmospheric exit is constrained to be $\geq 15,000$ feet per second.

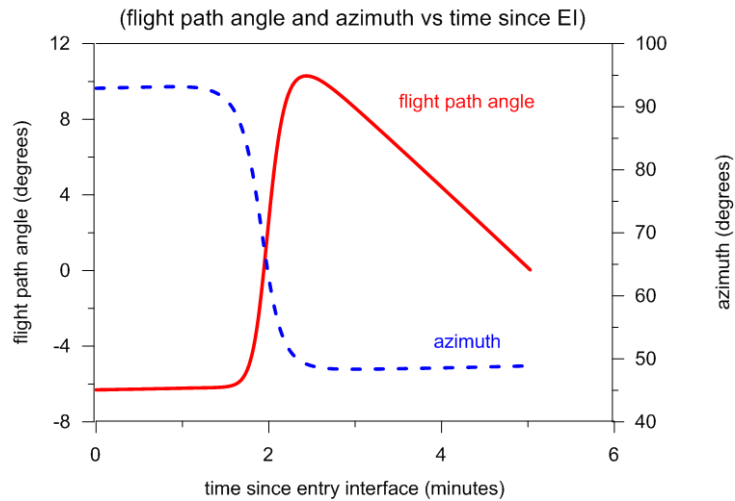
For this type of trajectory optimization make sure the orbital inclination at the atmospheric exit is not constrained by using the following statement in the input file

```
enforce an orbital inclination constraint (yes or no)  
no
```

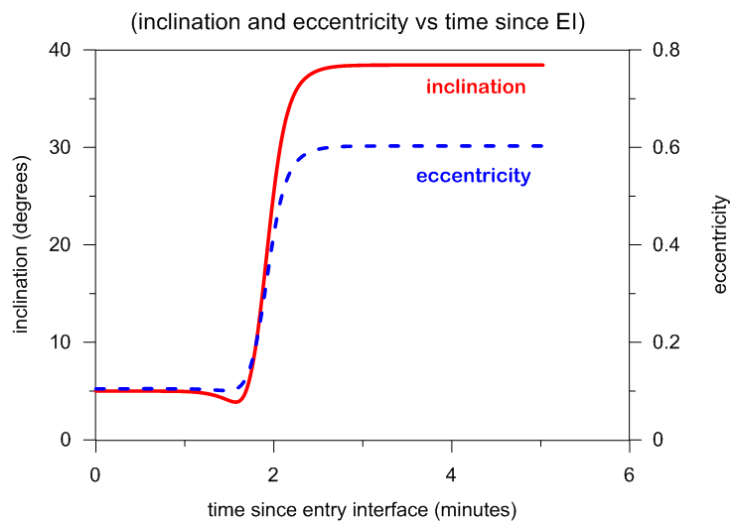
The following are plots of the important trajectory parameters for this example.



Aeroassist Trajectory Optimization Maximize orbital inclination change



Aeroassist Trajectory Optimization Maximize orbital inclination change



The following is the aeroassist program output for this example.

```

program aeroassist
=====

input file ==> leo2leo_max_inc.in

bounded entry conditions derived from deorbit maneuver

maximize orbital inclination change

orbital elements and state vector prior to deorbit impulse
-----

calendar date          January    1, 2001

```

```

universal time          00:00:00.000

      sma (nm)          eccentricity      inclination (deg)      argper (deg)
0.394392019016D+04    0.248334991895D-15    0.500000000000D+01    0.000000000000D+00

      raan (deg)        true anomaly (deg)      arglat (deg)          period (min)
0.431780632080D-14    0.300000000000D+02    0.300000000000D+02    0.103541236919D+03

      r-perigee (nm)    h-perigee (nm)          r-apogee (nm)        h-apogee (nm)
0.394392019016D+04    0.500000000000D+03    0.394392019016D+04    0.500000000000D+03

      rx (ft)           ry (ft)                  rz (ft)               rmag (ft)
0.207531855633D+08    0.119362626872D+08    0.104428766999D+07    0.239637145430D+08

      vx (fps)          vy (fps)                 vz (fps)              vmag (fps)
-0.121182361413D+05    0.209095296884D+05    0.182934680739D+04    0.242364722827D+05

```

orbital elements and state vector after deorbit impulse

```

-----
calendar date          January    1, 2001
universal time          00:00:00.000

      sma (nm)          eccentricity      inclination (deg)      argper (deg)
0.357017458724D+04    0.104685525536D+00    0.500000000000D+01    0.210000000000D+03

      raan (deg)        true anomaly (deg)      arglat (deg)          period (min)
0.456326147825D-14    0.180000000000D+03    0.300000000000D+02    0.891775123771D+02

      r-perigee (nm)    h-perigee (nm)          r-apogee (nm)        h-apogee (nm)
0.319642898432D+04    -0.247491205844D+03    0.394392019016D+04    0.500000000000D+03

      rx (ft)           ry (ft)                  rz (ft)               rmag (ft)
0.207531855633D+08    0.119362626872D+08    0.104428766999D+07    0.239637145430D+08

      vx (fps)          vy (fps)                 vz (fps)              vmag (fps)
-0.114664033296D+05    0.197848183550D+05    0.173094731598D+04    0.229328066592D+05

```

flight path coordinates at atmospheric entry

```

-----
altitude                400000.000000000      feet
velocity                24368.0971648879      feet/second
declination              4.17633335544806      degrees
longitude                16.1785108288942      degrees
azimuth                 92.9277990467808      degrees
flight path angle       -6.30689859387635      degrees

inertial fpa            -5.93056753615202      degrees

```

orbital elements and state vector at atmospheric entry

```

-----
calendar date          January    1, 2001
universal time          00:26:03.919

      sma (nm)          eccentricity      inclination (deg)      argper (deg)
0.357017458725D+04    0.104685525537D+00    0.500000000013D+01    0.20999999997D+03

      raan (deg)        true anomaly (deg)      arglat (deg)          period (min)
0.373173703543D-06    0.273322949484D+03    0.123322949481D+03    0.891775123775D+02

```

r-perigee (nm)	h-perigee (nm)	r-apogee (nm)	h-apogee (nm)
0.319642898433D+04	-.247491205835D+03	0.394392019018D+04	0.500000000016D+03
rx (ft)	ry (ft)	rz (ft)	rmag (ft)
-.117154107210D+08	0.177516413688D+08	0.155306738546D+07	0.213256568000D+08
vx (fps)	vy (fps)	vz (fps)	vmag (fps)
-.200622216447D+05	-.163311939918D+05	-.142879432473D+04	0.259083401194D+05

flight path coordinates at atmospheric exit

altitude	400000.000000000	feet
velocity	15000.000000000	feet/second
declination	8.99670339727738	degrees
longitude	29.5623003632225	degrees
azimuth	48.8880628810677	degrees
flight path angle	3.984582451546240E-002	degrees

orbital elements and state vector at atmospheric exit

calendar date January 1, 2001

universal time 00:31:05.462

sma (nm)	eccentricity	inclination (deg)	argper (deg)
0.218954752865D+04	0.602957760689D+00	0.384433270708D+02	0.194591552768D+03
raan (deg)	true anomaly (deg)	arglat (deg)	period (min)
0.126562332360D+03	0.179975688582D+03	0.145672413500D+02	0.428304598571D+02
r-perigee (nm)	h-perigee (nm)	r-apogee (nm)	h-apogee (nm)
0.869342853852D+03	-.257457733631D+04	0.350975220345D+04	0.658320132855D+02
rx (ft)	ry (ft)	rz (ft)	rmag (ft)
-.156695378708D+08	0.140757933205D+08	0.333485580891D+07	0.213256568000D+08
vx (fps)	vy (fps)	vz (fps)	vmag (fps)
-.743898214339D+04	-.105738521023D+05	0.974327143645D+04	0.161887659164D+05

aerodynamic characteristics

drag coefficient at aoa = 0 degrees 5.000000000000000E-002

drag coefficient at max L/D 0.100000000000000

lift coefficient at max L/D 0.188982236504614

maximum lift-to-drag ratio 1.88982236504614

APPENDIX E

De-orbit from a Circular Earth Orbit

The scalar magnitude of the single impulsive maneuver required to de-orbit a spacecraft from an initial circular orbit can be determined from the following expression

$$\Delta V = V_{c_e} \sqrt{\frac{1}{\tilde{r}}} \left\{ 1 - \frac{\sqrt{2(\tilde{r}-1)}}{\sqrt{\left(\frac{\tilde{r}}{\cos \gamma_e}\right)^2 - 1}} \right\} = V_{c_i} \left\{ 1 - \frac{\sqrt{2(\tilde{r}-1)}}{\sqrt{\left(\frac{\tilde{r}}{\cos \gamma_e}\right)^2 - 1}} \right\}$$

where

$$\tilde{r} = \frac{h_i + r_{eq}}{h_e + r_{eq}} = \frac{r_i}{r_e} = \text{radius ratio}$$

$$V_{c_e} = \sqrt{\frac{\mu}{(h_e + r_{eq})}} = \sqrt{\frac{\mu}{r_e}} = \text{local circular velocity at entry interface}$$

$$V_{c_i} = \sqrt{\frac{\mu}{(h_i + r_{eq})}} = \sqrt{\frac{\mu}{r_i}} = \text{local circular velocity of initial circular orbit}$$

γ_e = flight path angle at entry interface

h_i = altitude of initial circular orbit

h_e = altitude at entry interface

r_i = radius of initial circular orbit

r_e = radius at entry interface

r_{eq} = Earth equatorial radius

μ = Earth gravitational constant

This algorithm is described in the technical article, "Deboost from Circular Orbits", A. H. Milstead, *The Journal of the Astronautical Sciences*, Vol. XIII, No. 4, pp. 170-171, Jul-Aug., 1966. Additional information can be found in Chapter 5 of *Hypersonic and Planetary Entry Flight Mechanics* by Vinh, Busemann and Culp, The University of Michigan Press.

The true anomaly on the de-orbit trajectory at the entry interface θ_e can be determined from the following two equations

$$\sin \theta_e = \frac{\dot{r}}{e_d} \sqrt{\frac{a_d(1-e_d^2)}{\mu}}$$

$$\cos \theta_e = \frac{a_d(1-e_d^2)}{e_d r_e} - \frac{1}{e_d}$$

and the following four quadrant inverse tangent operation

$$\theta_e = \tan^{-1}(\sin \theta_e, \cos \theta_e)$$

where

e_d = eccentricity of the de-orbit trajectory

a_d = semimajor axis of the de-orbit trajectory

$$\dot{r} = -\sqrt{\frac{\mu[2a_d r_e - r_e^2 - a_d^2(1-e_d^2)]}{a_d r_e^2}}$$

The elapsed time-of-flight between perigee of the de-orbit trajectory and the entry true anomaly θ_e is given by

$$t(\theta_e) = \frac{\tau}{2\pi} \left[2 \tan^{-1} \left\{ \sqrt{\frac{1-e_d}{1+e_d}} \tan \frac{\theta_e}{2} \right\} - \frac{e_d \sqrt{1-e_d^2} \sin \theta_e}{1+e_d \cos \theta_e} \right]$$

In this equation τ is the Keplerian orbital period of the de-orbit trajectory and is equal to $2\pi\sqrt{a_d^3/\mu}$.

Therefore, the flight time between the de-orbit impulse and entry interface is given by

$$\Delta t = t(\theta_e) - t(180^\circ) = t(\theta_e) - \frac{\tau}{2}$$

Finally, the orbital speed at the entry interface V_e can be determined from

$$V_e = \sqrt{\frac{2\mu}{r_e} - \frac{\mu}{a_d}}$$

APPENDIX F

Typical Configuration File

The `aeroassist` computer program can read and use a user-defined configuration file. A description of each element in this file can be found in the **INSOCX** routine in section 6.2, *Subprograms for Optimal Control*, and the **INSNLP** routine in Section 2.2, *Subprograms for Optimization* of the `AMA_OC` user's manual. Please note that the `aeroassist` software can read and use a subset of the information in this file. For example, a subset configuration file might contain only the following information;

```
ODETOL=0.1D-06
INSNLP:IOFLAG=5
SOCOUT=I4K4
```

The following is a typical “full version” configuration file created during the execution of the `aeroassist` software.

```
AEQTOL=0.1000000000000000D-02
DTAUX=0.0000000000000000D+00
OBJCTL=0.1000000000000000D-04
ODETOL=0.1000000011686097D-06
PGDCTL=0.1000000000000000D-02
PRTMSD=0.1490116119384766D-07
PRTMXD=0.1000000000000000D-02
PRTSFD=0.1000000000000000D-04
QDRTOL=0.1000000000000000D-02
RESTOL=0.1000000000000000D-04
SMLTOL=0.1490116119384766D-10
TOLJSD=0.1000000000000000D-05
TOLM5A=0.1490116119384766D-07
TOLM5R=0.1490116119384766D-07
IDSCPH=0
IDSCND=0
IDSCVR=0
IDSCFN=0
IDTSFD=-1
IPFAUX=0
IPFSFD=0
IPRSFD=1
IPGRD=0
IPNLP=10
IPODE=0
IPUAUX=0
IPUOCP=6
IRSTRT=0
ISCALE=0
ISFHES=41
ISFINP=42
ISFRST=43
ISFSCL=44
ITSWCH=2
M5DTYP=0
MITODE=20
MTSWCH=-1
MXDATA=0
MXPARM=10
MXPCON=20
MXSTAT=20
MXTERM=50
NPTAUX=100
NSSWCH=-1
```

SOCOUT=A0B0C0D0E0F0G0H0I0J2K0L0M0N0O0P0Q0R0S1T0U0V0W0X0Y0Z0
SPRTHS=SPARSE
NLPALG=SNLPMN
NLPOMR=M
KEYDPL=.lueiLUE
RHSTMP=RHSTMPLT
RSTFIL=socx.restart
SCLFIL=scalewgt.fil
INSNLP:ALFLWR=0.000000000000000D+00
INSNLP:ALFUPR=0.100000000000000D+01
INSNLP:CONTOL=0.1490116119384766D-07
INSNLP:EPSRLF=0.1490116119384766D-07
INSNLP:OBJTOL=0.9999999747378752D-05
INSNLP:PGDTOL=0.100000000000000D-04
INSNLP:SLPTOL=0.900000000000000D+00
INSNLP:SFZTOL=0.100000000000000D-01
INSNLP:TOLFIL=0.200000000000000D+01
INSNLP:TOLKTC=0.1110953834938985D+26
INSNLP:TOLPVT=0.100000000000000D-02
INSNLP:IHESHN=0
INSNLP:IOFLAG=5
INSNLP:IOFLIN=-1
INSNLP:IOFMFR=0
INSNLP:IOFPAT=0
INSNLP:IOFSHR=0
INSNLP:IOFSRC=0
INSNLP:IPUDRF=0
INSNLP:IPUFZF=0
INSNLP:IPUMF1=11
INSNLP:IPUMF2=12
INSNLP:IPUMF3=13
INSNLP:IPUMF4=14
INSNLP:IPUMF5=15
INSNLP:IPUMF6=16
INSNLP:IPUMF7=17
INSNLP:IPUNLP=6
INSNLP:IPUSTF=0
INSNLP:IRELAX=1
INSNLP:ITDRQP=-1
INSNLP:ITFZQP=-1
INSNLP:ITLMAX=20
INSNLP:JACPRM=0
INSNLP:LYNFNC=0
INSNLP:LYNOUT=0
INSNLP:LYNPLT=0
INSNLP:LYNPNT=101
INSNLP:LYNVAR=0
INSNLP:MAXLYN=5
INSNLP:MAXNFE=50000
INSNLP:MNSAME=2
INSNLP:NEWTON=0
INSNLP:NITMAX=50000
INSNLP:NITMIN=0
INSNLP:NORMAL=0
INSNLP:ALGOPT=FM
INSNLP:KTOPTN=SMALL
INSNLP:QPOPTN=SPARSE
INSNLP:BIGCON=-0.100000000000000D+01
INSNLP:FEATOL=0.100000000000000D-01
INSNLP:PMULWR=0.100000000000000D+00
INSNLP:PTHOL=0.100000000000000D+02
INSNLP:RHOLWR=0.100000000000000D+03
INSNLP:IMAXMU=10
INSNLP:MUCALC=3
INSNLP:MXQPIT=1