Optimal Control of an Aerocapture Maneuver

Thomas Antony*

Purdue University, West Lafayette, Indiana 47907

In this project, optimal control theory is used to compute the maximum time of flight trajectory for an aerocapture maneuver. This objective function is an indirect way of specifying that we want the vehicle to fly higher in the atmosphere at a region of lower drag, rather than quickly diving deep into the atmosphere and incurring high heating rates and G-loads. This was seen as a way to satisfy the vehicle constraints on heat-rates and G-loading without explicitly adding path constraints to the problem. The total heating rate and G-loading of the lifting maneuver is compared to that of a ballistic trajectory of the same kind as well as for a trajectory optimized for minimum time of flight. The effects of increasing the Lift-to-Drag ratio of the vehicle is also examined.

Nomenclature

β	Ballistic coefficient, kg/m^2	C_d	Coefficient of Drag
$\dot{q}_{conv}$	Convective heat flux, W/m^2	H	Hamiltonian
$\dot{q}_{rad}$	Radiative heat flux, W/cm^2	h_{scale}	Scale height, m
$\frac{L}{D}$	Lift-to-Drag ratio	m	Mass, kg
γ	Flight path angle, rad	r	Radial position, m
μ	Standard gravitational parameter, m^3/s^2	R_0	Initial position, m
ρ_0	Atmospheric density at sea-level, kg/m^3	R_E	Radius of the Earth, m
θ	Downrange position, rad	r_n	Nose radius, m
a_G	Total acceleration/deceleration, G's	u	Lift modulation control
A_{ref}	Reference area, m^2	V	Velocity, m/s
C_l	Coefficient of Lift	V_0	Initial velocity, m/s

I. Introduction

AEROCAPTURE is a technique used to decelerate a spacecraft arriving at a planet at hyperbolic velocities, using the planet's atmosphere, in order to capture it into a closed orbit around it. Flying through the atmosphere causes heating and G-loading which are of concern, especially in the case of a crewed vehicle. An aerocapture maneuver can be done in two ways. It can be performed using only aerodynamic drag or a combination of both aerodynamic drag and lift.

In a ballistic entry trajectory, the vehicle does not (or cannot) use aerodynamic lift during the maneuver. In this case, it has no control over its trajectory once it enters the atmosphere, and the trajectory

*Graduate Student, School of Aeronautics and Astronautics, AIAA Student Member, tantony@purdue.edu

is determined entirely by the entry conditions (altitude, velocity and flight path angle). However, if the vehicle is capable of producing lift, this can be used to control the trajectory using lift modulation.[1,2] For example, it can be used to cause the vehicle to fly at a higher altitude (with lower atmospheric density) for a longer duration, which results in lower heating rates and G-loading. It also provides a certain degree of control over the trajectory in flight. This is called lifting entry. Using aerodynamic lift provides us with a control that can be used to optimize the trajectory. In this project, we examine how the lift can be used to prolong atmospheric flight, thereby indirectly extending the heat-pulse and decreasing the peak heat flux and G-loading on the spacecraft.

II. Problem Statement and Assumptions

1. A SpaceX Dragon class vehicle with the specifications in Table 1 will be used for this analysis. The vehicle is trimmed to an angle-of-attack required for the given $\frac{L}{D}$. Lift modulation is used as the control.
2. A spherical Earth gravity model with an exponential atmosphere (Table 2) is assumed.
3. The entry conditions correspond to a hypothetical incoming Mars-Earth trajectory with an entry velocity of 14.2 km/s at 120 km altitude. Targeted exit condition is a velocity of 10.5 km/s at 120 km altitude that corresponds to a 16 hour orbit ($sma = 32057.5km$).
4. A vehicle-centric polar coordinate system and 2-DOF dynamic model [3, p.100] is used.
5. Convective and radiative heating models described by Sutton and Graves[4] and Tauber[5] respectively are used for heating analysis.
6. A continuation method in which the boundary conditions are gradually changed to the actual terminal values will be used to help with convergence.[6]

Table 1: Vehicle Model[7]

Parameter	Value
m	5000 kg
C_d	1.30
A_{ref}	10.2 m^2
$\frac{L}{D}$	0.18
r_n	3.0 m

Table 2: Gravity and Atmosphere Model[8]

Parameter	Value
μ	3.986×10^{14} m^3/s^2
R_E	6.3781×10^6 m
ρ_0	1.217 kg/m^3
h_{scale}	8500 m

The optimal control problem is formally stated as follows:

$$\text{Max } J = t_f$$

Subject to :

$$\dot{r} = V\sin(\gamma) \tag{1a}$$

$$\dot{\theta} = \frac{V\cos(\gamma)}{r} \tag{1b}$$

$$\dot{V} = -\frac{\rho V^2}{2\beta} - \frac{\mu\sin(\gamma)}{r^2} \tag{1c}$$

$$\dot{\gamma} = \frac{V\cos(\gamma)}{r} + \frac{\rho V\frac{L}{D}u}{2\beta} - \frac{\mu\cos(\gamma)}{(Vr^2)} \tag{1d}$$

$$r(t_0) = R_E + 120 \times 10^3 \text{ m} \tag{2a}$$
$$v(t_0) = 14.2 \times 10^3 \text{ m/s} \tag{2b}$$
$$\theta(t_0) = 0 \tag{2c}$$
$$r(t_f) = R_E + 120 \times 10^3 \text{ m} \tag{2d}$$
$$v(t_f) = 10.5 \times 10^3 \text{ m/s} \tag{2e}$$
$$t_0 = 0 \tag{2f}$$

where,

$$\rho = \rho_0 \exp(-(r - R_E)/h_{scale})$$
$$\beta = \frac{m}{C_d A_{ref}}$$
$$-1 \leq u \leq 1$$

III. Analysis

The state variables are scaled to help with convergence later in the process,

$$\begin{aligned} \bar{r} &= \frac{r}{R_0} \\ \bar{V} &= \frac{V}{V_0} \end{aligned} \tag{3}$$

where R_0 and V_0 are the initial values of r and V.
Let

$$K_1 = \frac{V_0}{2\beta}, K_2 = \frac{\mu}{V_0 R_0^2}, K_3 = \frac{V_0}{R_0}, K_4 = \frac{R_0}{h_{scale}} \tag{4}$$

The scaled equations of motion can be obtained by substituting (3) and (4) in (1).

$$\frac{d\bar{r}}{dt} = K_3 \bar{V} \sin(\gamma) \tag{5a}$$
$$\frac{d\theta}{dt} = \frac{K_3 \bar{V} \cos(\gamma)}{\bar{r}} \tag{5b}$$
$$\frac{d\bar{V}}{dt} = -K_1 \rho \bar{V}^2 - \frac{K_2 \sin(\gamma)}{\bar{r}^2} \tag{5c}$$
$$\frac{d\gamma}{dt} = \frac{K_3 \bar{V} \cos(\gamma)}{\bar{r}} + K_1 \rho \bar{V} u - \frac{K_2 \cos(\gamma)}{(\bar{V}\bar{r}^2)} \tag{5d}$$

where $\rho = \rho_0 \exp(-(\bar{r}R_0 - R_E)/h_{scale})$

Applying Euler Lagrange equation[9] to Eqs. (5)

$$\begin{aligned} H =& L + \lambda^T f \\ =& \bar{\lambda_r}[K_3 \bar{V} \sin(\gamma)] + \bar{\lambda_\theta}[\frac{K_3 \bar{V} \cos(\gamma)}{\bar{r}}] + \bar{\lambda_V}[-K_1 \rho \bar{V}^2 - \frac{K_2 \sin(\gamma)}{\bar{r}^2}] \\ &+ \bar{\lambda_\gamma}[\frac{K_3 \bar{V} \cos(\gamma)}{\bar{r}} + K_1 \frac{L}{D} \rho \bar{V} u - \frac{K_2 \cos(\gamma)}{(\bar{V} \bar{r}^2)}] \end{aligned} \tag{6}$$

$$\begin{aligned} \frac{d\bar{\lambda_r}}{dt} =& -\frac{\partial H}{\partial \bar{r}} \\ =& \bar{\lambda_\theta}[\frac{K_3 \bar{V} \cos(\gamma)}{\bar{r}^2}] - \bar{\lambda_V}[K_1 K_4 \rho \bar{V}^2 + \frac{2K_2 \sin(\gamma)}{\bar{r}^3}] - \bar{\lambda_\gamma}[-K_1 K_4 \frac{L}{D} \rho u \bar{V} + \frac{2K_2 \cos(\gamma)}{\bar{r}^3 \bar{V}} - \frac{K_3 \bar{V} \cos(\gamma)}{\bar{r}^2}] \end{aligned} \tag{7a}$$

$$\begin{aligned} \frac{d\bar{\lambda_\theta}}{dt} =& -\frac{\partial H}{\partial \theta} \\ =& 0 \end{aligned} \tag{7b}$$

$$\begin{aligned} \frac{d\bar{\lambda_V}}{dt} =& -\frac{\partial H}{\partial \bar{V}} \\ =& -\bar{\lambda_r}[K_3 \sin(\gamma)] - \bar{\lambda_\theta}[\frac{K_3 \cos(\gamma)}{\bar{r}}] + \bar{\lambda_V}[2K_1 \rho \bar{V}] - \bar{\lambda_\gamma}[K_1 \frac{L}{D} \rho u + \frac{K_3 \cos(\gamma)}{\bar{r}} + \frac{K_2 \cos(\gamma)}{\bar{r}^2 \bar{V}^2}] \end{aligned} \tag{7c}$$

$$\begin{aligned} \frac{d\bar{\lambda_\gamma}}{dt} =& -\frac{\partial H}{\partial \gamma} \\ =& -\bar{\lambda_r}[K_3 \bar{V} \cos(\gamma)] + \bar{\lambda_\theta}[\frac{K_3 \bar{V} \sin(\gamma)}{\bar{r}}] + \bar{\lambda_V}[\frac{K_2 \cos(\gamma)}{\bar{r}^2}] - \bar{\lambda_\gamma}[\frac{K_2 \sin(\gamma)}{\bar{V} \bar{r}^2} - \frac{K_3 \bar{V} \sin(\gamma)}{\bar{r}}] \end{aligned} \tag{7d}$$

u corresponds to the fraction of aerodynamic lift acting upwards. In reality, this is to be implemented using bank angle modulation. Negative values of u correspond to the lift vector being directed downwards.

Since the Hamiltonian is linear w.r.t the control, Pontryagin's Minimum Principle[9] is used to obtain the control law:

$$u(t) = \begin{cases} -1 & \text{when } H_2 > 0 \\ \text{undefined} & \text{when } H_2 = 0 \\ +1 & \text{when } H_2 < 0 \end{cases} \tag{8}$$

$$\text{where } H_2 = \frac{\partial H}{\partial u} = K_1 \bar{\lambda_\gamma} \rho \bar{V}$$

The singular case (when $H_2 = 0$) was ignored in this analysis as it occurred only at two points in the beginning and the end of the trajectory and hence doesn't affect the overall trajectory.
The transversality condition is :

$$H(t_f) dt_f + \lambda(t_f)^T d\vec{x}_f + \lambda(t_0)^T d\vec{x}_0 + dg = 0 \tag{9}$$

From (2), $dr_0 = dv_0 = dr_f = dv_f = 0, dg = -dt_f$. Substituting these values in (9), we get

$$\lambda_\gamma(0) = 0 \tag{10a}$$

$$\lambda_\theta(t_f) = 0 \tag{10b}$$

$$\lambda_\gamma(t_f) = 0 \tag{10c}$$

$$(H(t_f) - 1)dt_f = 0 \Rightarrow H(t_f) - 1 = 0 \tag{10d}$$

From the boundary conditions (10) and costate equations (7), it can be seen that λ_θ is zero throughout the trajectory and can hence be ignored.

The Two Point Boundary Value Problem (TPBVP) is obtained from (5), (7), (8), (2) and (10).

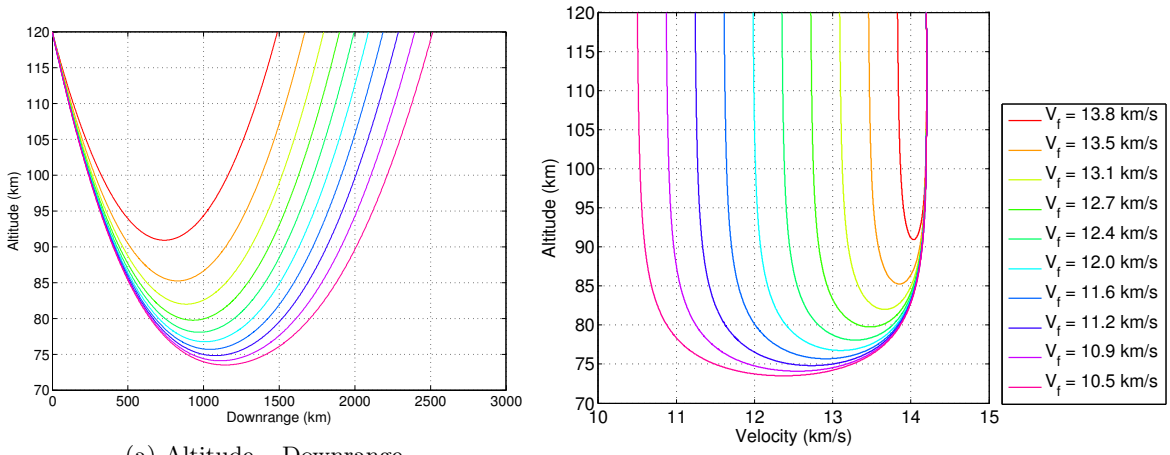
$$\begin{aligned}
\frac{d\bar{r}}{dt} =& K_3\bar{V}\sin(\gamma) \\
\frac{d\theta}{dt} =& \frac{K_3\bar{V}\cos(\gamma)}{\bar{r}} \\
\frac{d\bar{V}}{dt} =& -K_1\rho\bar{V}^2 - \frac{K_2\sin(\gamma)}{\bar{r}^2} \\
\frac{d\gamma}{dt} =& \frac{K_3\bar{V}\cos(\gamma)}{\bar{r}} + K_1\frac{L}{D}\rho\bar{V}u - \frac{K_2\cos(\gamma)}{(\bar{V}\bar{r}^2)} \\
\frac{d\bar{\lambda_r}}{dt} =& -\bar{\lambda_V}[K_1K_4\rho\bar{V}^2 + \frac{2K_2\sin(\gamma)}{\bar{r}^3}] - \bar{\lambda_\gamma}[-K_1K_4\frac{L}{D}\rho u\bar{V} + \frac{2K_2\cos(\gamma)}{\bar{V}\bar{r}^3} - \frac{K_3\bar{V}\cos(\gamma)}{\bar{r}^2}] \\
\frac{d\bar{\lambda_V}}{dt} =& -\bar{\lambda_r}[K_3\sin(\gamma)] + \bar{\lambda_V}[2K_1\rho\bar{V}] - \bar{\lambda_\gamma}[K_1\frac{L}{D}\rho u + \frac{K_3\cos(\gamma)}{\bar{r}} + \frac{K_2\cos(\gamma)}{\bar{r}^2\bar{V}^2}] \\
\frac{d\bar{\lambda_\gamma}}{dt} =& -\bar{\lambda_r}[K_3\bar{V}\cos(\gamma)] + \bar{\lambda_V}[\frac{K_2\cos(\gamma)}{\bar{r}^2}] - \bar{\lambda_\gamma}[\frac{K_2\sin(\gamma)}{\bar{V}\bar{r}^2} - \frac{K_3\bar{V}\sin(\gamma)}{\bar{r}}] \\
\bar{r}(t_0) =& 1.0 \\
\bar{V}(t_0) =& 1.0 \\
\theta(t_0) =& 0 \\
\bar{r}(t_f) =& 1.0 \\
\bar{V}(t_f) =& \frac{10.5\times10^3}{V_0} \\
t_0 =& 0 \\
\bar{\lambda_\gamma}(0) =& \bar{\lambda_\gamma}(t_f) = 0 \\
H(t_f) - 1 =& 0 \\
u(t) =& \begin{cases} -1 & \text{when } H_2 > 0 \\ \text{undefined} & \text{when } H_2 = 0 \\ +1 & \text{when } H_2 < 0 \end{cases} \\
\text{where } H_2 =& K_1\bar{\lambda_\gamma}\rho\bar{V}, \\
\rho =& \rho_0\exp(-(\bar{r}R_0 - R_E)/h_{scale}), \\
K_1 =& \frac{V_0}{2\beta}, K_2 = \frac{\mu}{V_0R_0^2}, K_3 = \frac{V_0}{R_0}, K_4 = \frac{R_0}{h_{scale}} \\
R_0 =& R_E + 120\times10^3 \text{ m} \\
V_0 =& 14.2\times10^3 \text{ m/s}
\end{aligned}$$

The 7 equations and 9 boundary conditions form a well-defined TPBVP.

IV. Numerical Results

IV.A. Continuation Process

The TPBVP in Eq. (11) was solved using MATLAB's *bvp4c* solver to obtain the following results. The problem cannot be solved directly in one step as *bvp4c* fails to converge and hence a continuation process[6] is required. In order to do this, the targeted final velocity is initially set to a higher value (e.g. 90% of V_0) and then it is decreased down to its desired value in a series of steps, with the solution for each iteration serving as the initial guess for the next iteration. This way, we initially solve a simpler problem, and then gradually change the problem until we are targeting the desired conditions.



(a) Altitude – Downrange

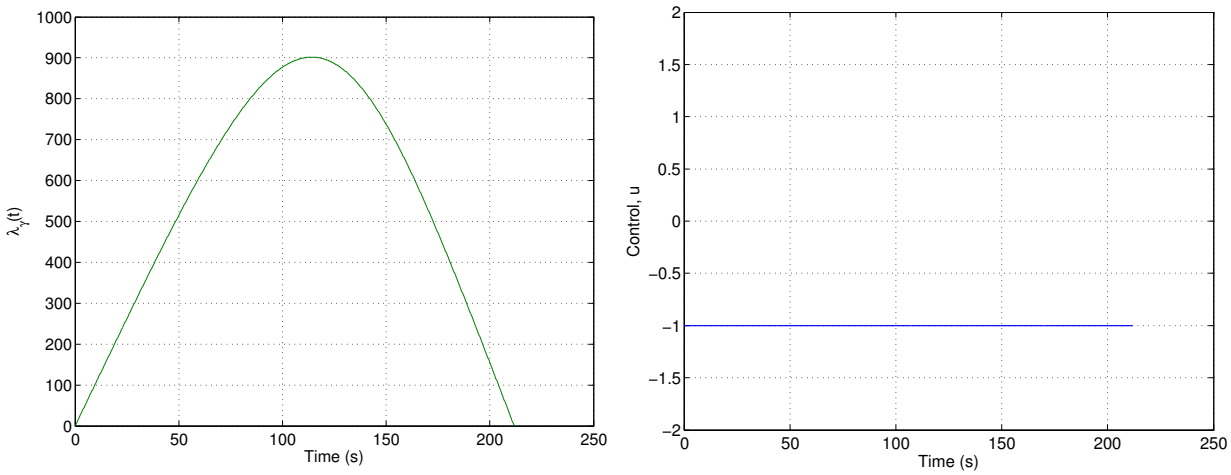
(b) Altitude – Velocity

Figure 1: Continuation process for solving the TPBVP

Fig. 1 shows how the trajectory changes as the targeted final velocity is decreased in each step, over 10 steps until finally reaching the desired value of 10.5 km/s. Solving the problem for a short trajectory makes the problem comparatively insensitive to the initial guess and converges without any difficulty. Subsequent iterations using the prior solutions as the initial guess are much faster than the first step when the "step-size" (or change in the boundary conditions) is suitably small. 10 steps (or a change of around 370m/s in V_f in each iteration) was found to be the lower limit for this particular case. Fewer number of steps resulted in *bvp4c* failing to converge at some point during the process.

IV.B. Switching Function and Control History

Since the problem results in a bang-bang style control, the switching function determines the direction of the lift vector during flight. This is a function of ρ, V and λ_γ which reduces to being just λ_γ since all the other variables in it are positive on a nominal trajectory. From Fig. 2a, it can be seen that the switching function does not change sign and remains positive for the entire trajectory. This means that the optimum trajectory involves flying with the lift vector pointed downwards (towards the surface of the planet) for the whole maneuver (Fig. 2b). This makes sense because, the vehicle will tend to move away from the planet very rapidly because of its hyperbolic incoming velocity, and the only thing that holds it inside the atmosphere for a longer period of time (and at a higher altitude than a ballistic trajectory) is the aerodynamic lift. If it had no lift, the vehicle would have had to dive deeper into the atmosphere at a steeper angle to achieve the same velocity reduction. This is further illustrated in the next section in Fig. 3a and Fig. 3b.



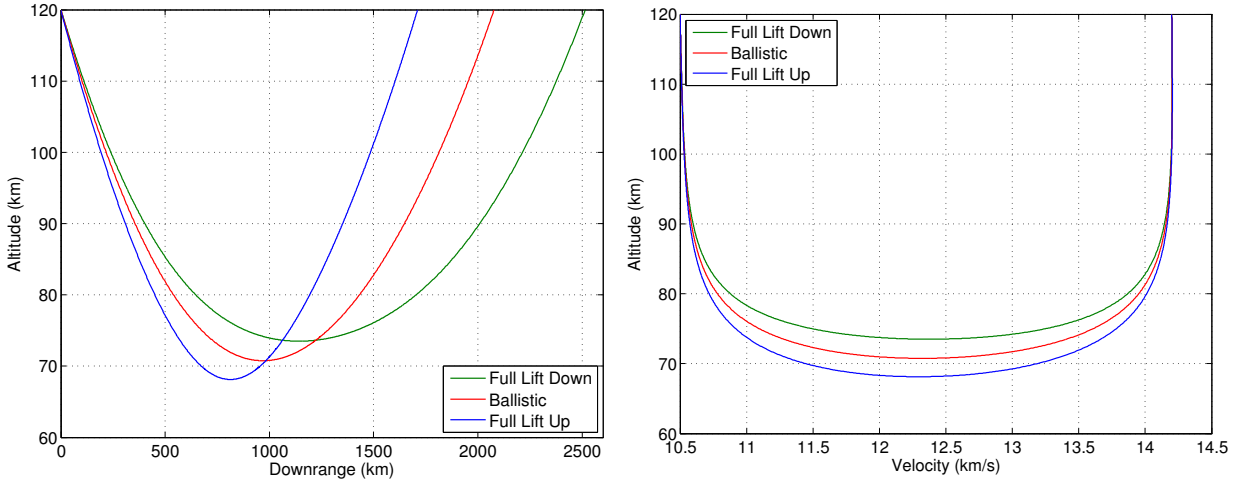
(a) Switching Function

(b) Control History

Figure 2: Bang-bang Control

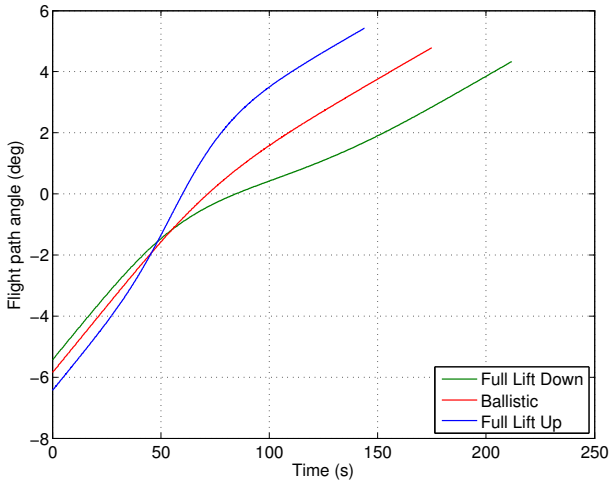
IV.C. Trajectory Modes and Entry Corridor

Fig. 3 shows the trajectories that satisfy the boundary conditions when flying full lift up, full lift down and ballistic (no lift). It can be seen that the entry angle for the ballistic trajectory is much steeper than that for flying full lift down. Flying full lift up results in an even steeper trajectory. This trajectory was obtained by optimizing for minimum time of flight. This is an example of how *not* to use lift for aerocapture. This will be illustrated further in the next section when examining the heating rate and G-loading.



(a) Altitude – Downrange

(b) Altitude – Velocity



(c) Flight Path Angle, $\gamma(t)$

Figure 3: Comparison of three entry modes – Full Lift Down, Ballistic and Full Lift Up

Another advantage of using lift during entry is a wider "entry corridor". In the case of a ballistic entry, the only parameter that controls the entire trajectory is the entry flight path angle. If this angle is too shallow, it will result in the spacecraft skipping out of the atmosphere on a hyperbolic trajectory, whereas too steep an entry will result in a direct entry that incurs very high heating and G-loads (essentially burning up the spacecraft). The success or failure of the maneuver is highly dependent on the accuracy of the interplanetary navigation system that delivers the spacecraft to the entry interface at the correct angle.

However, when using lifting entry, the system is more forgiving of errors in the entry angle. Since we have the ability to modulate the lift to change the trajectory, we can, to some degree, correct for errors in the initial entry conditions while in flight. For example, when flying a lifting entry we have the capability of flying any of the entry angles between the green and blue lines on Fig. 3c depending on the amount of lift used. This control authority gives us a much wider entry corridor[1] and this also gives the option of using closed loop guidance in flight.

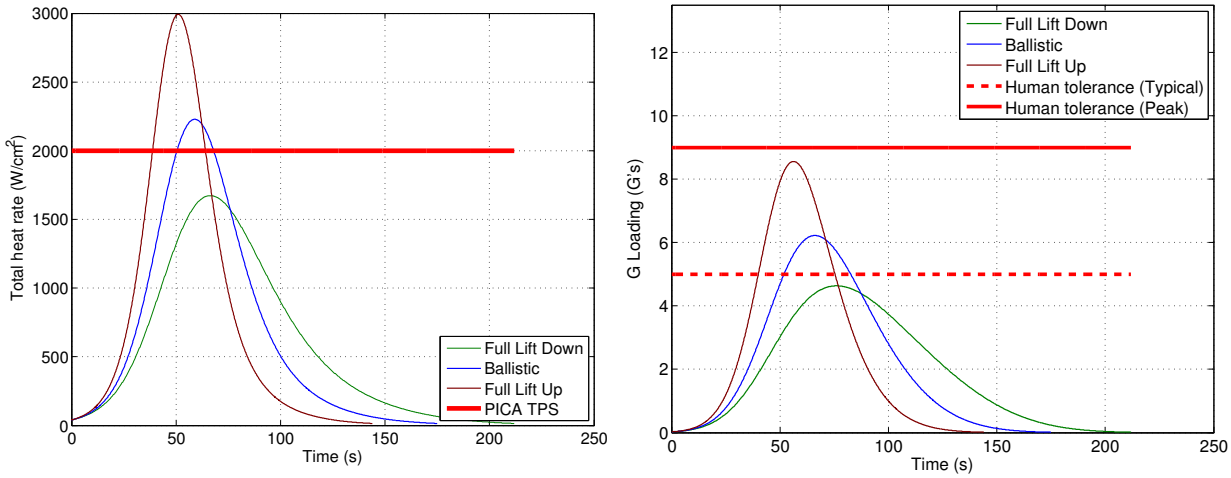
IV.D. Heat Rate and G-Loading Analysis

Heating in hypersonic entry vehicles happen mainly in two ways, convective and radiative heating. At hyperbolic velocities, as is the case in this problem, radiative heating plays a more dominant role. For analyzing the heating of the spacecraft, Sutton and Graves' convective heating model[4] and Tauber's[5] radiative heating model were used. Both models give equations for calculating the heating at the stagnation point of an axisymmetric body of a given nose radius. The equations are as follows:

$$\begin{aligned} \dot{q}_{conv} =& k\sqrt{\frac{\rho}{r_n}}V^3 \\ \text{where } k =& \ 1.74153 \times 10^{-4}\text{kg}^{0.5}\text{m}^{-2} \text{ for Earth,} \\ & r_n \rightarrow \text{ Nose radius of spacecraft} \end{aligned} \tag{12}$$

$$\begin{aligned} \dot{q}_{rad} =& C \cdot (r_n)^a \cdot \rho^b \cdot f(V) \\ \text{where } C =& 4.736 \times 10^4, a = 1.072 \times 10^6 V^{-1.88}\rho^{-0.325}, b = 1.22 \end{aligned} \tag{13}$$

The values for $f(V)$ where obtained from a curve fit of data from Ref. 5, where it is defined for velocities from 9000 m/s to 16000 m/s. $\dot{q}_{conv}$ is obtained in W/m^2 and $\dot{q}_{rad}$ is obtained in W/cm^2. Total heat rate is calculated by adding $\dot{q}_{conv}$ and $\dot{q}_{rad}$.



(a) Total Heat Rate

(b) G-Loading

Figure 4: Heat Rate and G-Loads

The Thermal Protection System (TPS) on the SpaceX Dragon is made using PICA-X, a material derived from NASA's PICA (Phenolic Impregnated Carbon Ablator). It is capable of handling heat-rates of up to 2000 W/cm^2.[10]

G-loading is calculated as the resultant acceleration from all aerodynamic forces on the body. The

$$a_G = \frac{1}{2}\frac{\rho V^2 A_{ref}\sqrt{{C_l}^2 + {C_d}^2}}{m \cdot 9.80665} \tag{14}$$

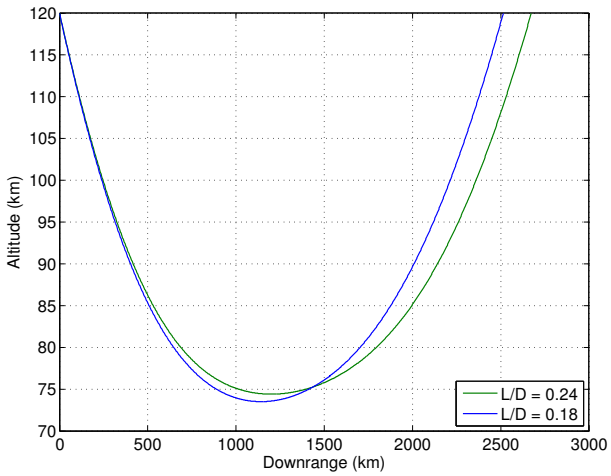
Fig. 4 compares the heat rate and G-loading for ballistic and lifting entry (full lift down). It is seen that the heat rate for the ballistic trajectory is much higher. This is due to the fact that it dives deeper into the atmosphere. The G-loading is also higher in the case of ballistic entry because of the higher drag caused by flying in lower altitudes. Human tolerance to G-loading varies depending on the level of training and the duration of exposure.[11] Persistent G-loads above 5G's can cause loss of consciousness. Astronauts weakened after a long-term zero-G cruise will be even more susceptible to these effects. It can be seen that utilizing aerodynamic lift for re-entry is very beneficial in this context.

However, optimizing for minimum time of flight makes the vehicle fly with the lift vector pointed up, which requires it to dive deeper into the atmosphere and incur even higher heat rates and G-loading.

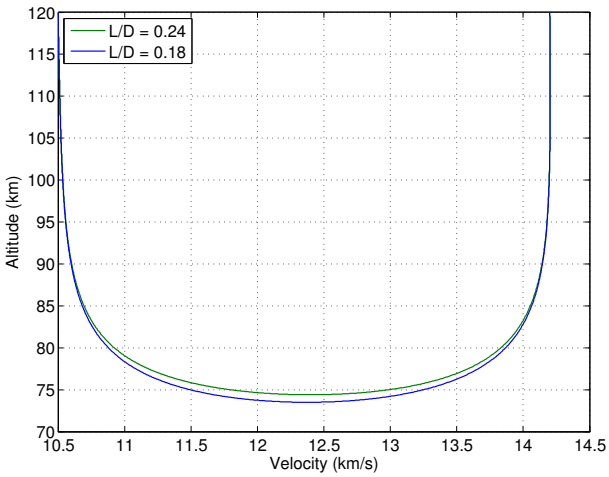
IV.E. Effect of Using More Lift

Grover et al.[12] mentions that it is possible to trim the SpaceX Dragon vehicle for a higher Lift-to-Drag ratio of 0.24. The following analysis was performed by assuming that C_d remains the same whereas $\frac{L}{D}$ was changed to 0.24. The effect of increasing the lift coefficient of the vehicle on the trajectory is examined in Fig. 5.

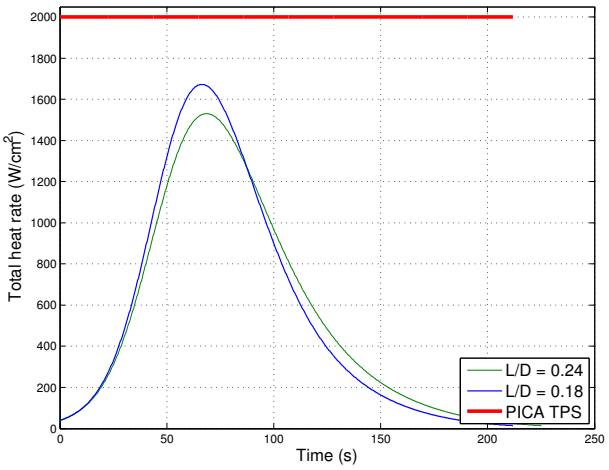
It was found that utilizing a higher lift results in lower heating rates and G-loading since the vehicle will be capable of flying at a higher altitude and still get captured into a closed orbit.



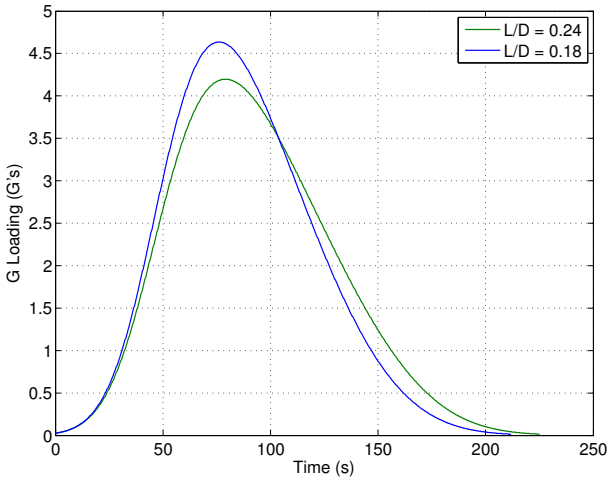
(a) Altitude – Downrange



(b) Altitude – Velocity



(c) Total Heat Rate



(d) G-Loading

Figure 5: Trajectory for $\frac{L}{D} = 0.24$ vs. $\frac{L}{D} = 0.18$

V. Conclusions and Future Work

1. Optimizing an aerocapture maneuver for maximum time of flight has the indirect result of reducing the peak heating rate as well as the G-Loading on the vehicle. The optimum control was found to be to fly full lift down for the entire duration of the trajectory. Conversely, optimizing for minimum time of flight resulted in a trajectory with heat flux and G-loading higher than that of a ballistic trajectory.

2. The vehicle is able to stay at a higher altitude at a region of lower atmospheric density for a longer period of time when utilizing lift. Consequently, the peak heat flux and G-loading on the spacecraft is lower than when lift is not utilized (ballistic entry).

3. The use of lift provides control authority that can be used to correct errors in entry conditions to a certain degree.

4. Using a spacecraft with a higher Lift-to-Drag ratio results in even better results in terms of heat-rate and G-loading.

Future expansions to this work can include the following :

- Extend the problem to investigate the descent and landing trajectory from the captured orbit.
- Use 3-DOF dynamic system to model cross-range flight from banking.
  - Target orbits of specified inclination.
  - Target specific energy changes and turn angles (Aero-gravity assist[13]), possibly using higher performance vehicles.

Appendix

MATLAB Code

aeroCaptureMain.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Program : aeroCaptureMain.m                                                %%
%%                                                                           %%
%%    This program uses bvp4c to solve the boundary value problem            %%
%% arising when optimizing an aero-capture trajectory for maximum            %%
%% time of flight.                                                           %%
%%                                                                           %%
%% Project : Optimal Control of an Aerocapture Maneuver                      %%
%% AAE 508 - Spring 2014                                                     %%
%%                                                                           %%
%% Author  : Thomas Antony                                                   %%
%% Purdue University                                                         %%
%% 28-Apr-2014                                                               %%
%%                                                                           %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear;

% The ODE and BC function to be used in this problem
odefn = @aeroCaptureOde;
bcfn  = @aeroCaptureBC;

% Constants
const.mass = 5000;          % Mass of vehicle, kg
const.Cd = 1.30;            % Coefficient of Drag
const.Aref = 10.2;          % Reference area, m^2
const.LD = 0.0;             % Lift-to-drag Ratio
const.beta0 = const.mass/(const.Cd*const.Aref); % Ballistic Coefficient

const.mu = 3.986e5*1e9;     % Standard gravitational parameter, m^3/s^2
const.re = 6378000;         % Radius of planet, m
const.rho0 = 1.217;         % Density of atmosphere at sea-level, kg/m^3
```

```
const.h_scale = 8.50e3;     % Scale height of atmosphere, m

const.rn = 3;               % Nose radius (for heating), m
const.k = 1.74153e-4;       % Sutton-graves heating constant for Earth, kg^(1/2) m^-2
const.g  = 9.80665;         % One Earth-G, m/s^2

numCases = 10;              % Number of steps used to solve problem

% Scaling factors
R0 = const.re+120e3;         % Scaling parameter for position
V0 = 14200;                 % Scaling parameter for velocity
scale = [R0;V0];

r0 = (const.re + 120e3)/R0; % Scaled initial position
v0 = 14200/V0;              % Scaled initial velocity
theta0 = 0*pi/180;          % Initial longitude (downrange position), rad

rF = (const.re + 120e3)/R0; % Scaled final position
vF = 11.1e3/V0;              % Scaled final velocity

tfGuess = 20;               % Guess for time-of-flight for first iteration

% Guess for initial and final states
x0Guess = [r0 theta0 v0 -1*pi/180 0 0 10]';
dvF = (vF-v0)/numCases;     % Change in velocity in one iteration
xFGuess = [rF 0.03 v0+dvF 1*pi/180 0 0 10]';

% Non-dimensional time vector
tau = linspace(0,1,100)';

% Create initial guess structure for bvp4c
solInit = bvpinit(tau,x0Guess,[tfGuess]);

% These will be used to "shift" the boundary conditions
% in each iteration
x0Guess = x0Guess(1:4,1);
xFGuess = xFGuess(1:4,1);

solGuess = solInit;
% Solve the problem using continuation
for i=1:numCases
    fprintf('Continuation %d/%d ... \n',i,numCases);
    fprintf('Targeting %f km/s ... \n',xFGuess(3)*V0/1000);
```

```
    bvpOptions = bvpset('AbsTol',1e-4,'RelTol',1e-4,'Stats','on','NMax',100000);
    % Call bvp4c to solve the BVP for given boundary conditions
    sol = bvp4c(odefn,bcfn,solGuess,bvpOptions,const,scale,x0Guess,xFGuess);
    solGuess = sol;

    % Change the targeted exit-velocity
    xFGuess(3) = xFGuess(3) + dvF;

    % Save the solution in output structure
    out.CONT(i).sol = sol;
end

% Save constants and scaling parameters in output
out.const = const;
out.scale = scale;

% Write output to file
save('result.mat','out');
```

aeroCaptureOde.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Program : aeroCaptureOde.m                                                  %%
%%                                                                            %%
%% ODE file for bvp4c                                                         %%
%%                                                                            %%
%% Project : Optimal Control of an Aerocapture Maneuver                       %%
%% AAE 508 - Spring 2014                                                      %%
%%                                                                            %%
%% Author  : Thomas Antony                                                    %%
%% Purdue University                                                          %%
%% 28-Apr-2014                                                                %%
%%                                                                            %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [ dXdTau ] = aeroCaptureOde(tau, X, p, const, scale, varargin)

    % Constants
    beta0 = const.beta0;        % Ballistic coefficient
    LD = const.LD;              % Lift-to-Drag Ratio
    mu = const.mu;              % Standard gravitiational parameter
    re = const.re;              % Radius of planet
    rho0 = const.rho0;          % Atmospheric density at sea-level
    h_scale = const.h_scale;    % Scale height of atmosphere

    % Actual Time of flight
```

```
tf = abs(p(1));

R0 = scale(1); V0 = scale(2);

% States and costates
r = X(1); theta = X(2); V = X(3); gam = X(4);
lamR = X(5); lamV = X(6); lamGAM = X(7);

% Scaling factors
R0 = scale(1);
V0 = scale(2);

% Scaling constants
K1 = V0/(2*beta0);
K2 = mu/(V0*R0^2);
K3 = V0/R0;
K4 = R0/h_scale;

% Scaled density
rho = rho0*exp(-(r*R0-re)/h_scale);

% Switching function for control law
H2 = K1*LD*rho*V*lamGAM;
if H2 < 0
    u = 1;
elseif H2 > 0
    u = -1;
else
    % Ignore the singular case since it is instantaneous
    u = 0;
end

% Dynamic equations
rDot     = K3*V*sin(gam);
thetaDot = K3*V*cos(gam)/r;
VDot     = -K1*rho*V^2 - K2*sin(gam)/r^2;
gamDot   = K3*V*cos(gam)/r + K1*LD*rho*V*u - K2*cos(gam)/(V*r^2);

% Costate equations
lamRDot  = -lamV*(K1*K4*rho*V^2 + 2*K2*sin(gam)/r^3) ...
           -lamGAM*(-K1*K4*LD*rho*u*V + 2*K2*cos(gam)/(V*r^3) - K3*V*cos(gam)/r^2);
lamVDot  = -lamR*(K3*sin(gam))...
           +lamV*(2*K1*rho*V)...
           -lamGAM*(K1*LD*rho*u + K3*cos(gam)/r + K2*cos(gam)/(r^2*V^2));
```

```
    lamGAMDot= -lamR*(K3*V*cos(gam))...
               +lamV*(K2*cos(gam)/r^2)...
               +lamGAM*(K2*sin(gam)/(V*r^2) - K3*V*sin(gam)/r);

    % Multiply with tf (time-scaling factor)
    dXdTau = tf*[rDot; thetaDot; VDot; gamDot; lamRDot; lamVDot; lamGAMDot];
end
```

aeroCaptureBC.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Program : aeroCaptureBC.m                                                 %%
%%                                                                          %%
%% Boundary condition file for bvp4c                                         %%
%%                                                                          %%
%% Project : Optimal Control of an Aerocapture Maneuver                      %%
%% AAE 508 - Spring 2014                                                     %%
%%                                                                          %%
%% Author  : Thomas Antony                                                   %%
%% Purdue University                                                         %%
%% 28-Apr-2014                                                               %%
%%                                                                          %%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [zero_vec] = aeroCaptureBC(YL,YR,p,const,scale,x0,xf)

    % Constants
    beta0 = const.beta0;        % Ballistic coefficient
    LD = const.LD;              % Lift-to-Drag Ratio
    mu = const.mu;              % Standard gravitiational parameter
    re = const.re;              % Radius of planet
    rho0 = const.rho0;          % Atmospheric density at sea-level
    h_scale = const.h_scale;    % Scale height of atmosphere

    % Actual time-of-flight
    tf = abs(p(1));

    % Right endpoint
    % States
    r = YR(1,1);
    theta = YR(2,1);
    V = YR(3,1);
    gam = YR(4,1);

    % Costates
    lamR = YR(5,1);
```

```
    lamV = YR(6,1);
    lamGAM = YR(7,1);

    % Scaling factors
    R0 = scale(1);
    V0 = scale(2);

    % Scaling constants
    K1 = V0*LD/(2*beta0);
    K2 = mu/(V0*R0^2);
    K3 = V0/R0;
    K4 = R0/h_scale;

    % Scaled density
    rho = rho0*exp(-(r*R0-re)/h_scale);

    % Switching function for control law
    H2 = K1*rho*V*lamGAM;
    if H2 < 0
        u = 1;
    elseif H2 > 0
        u = -1;
    else
        % Ignore the singular case since it is instantaneous
        u = 0;
    end

    % Hamiltonian at tf
    H_tf = lamR*(K3*V*sin(gam)) + lamV*(-K1*rho*V^2 -K2*sin(gam)/r^2) ...
          +lamGAM*(K3*V*cos(gam)/r + K1*r*V*u - K2*cos(gam)/(V*r^2));

    zero_vec = ([...
        YL(1,1) - x0(1);    % r0
        YR(1,1) - xf(1);    % rF
        YL(2,1) - x0(2);    % theta0
        YL(3,1) - x0(3);    % v0
        YR(3,1) - xf(3);    % vF
        YL(7,1) - 0;        % lamGAM0
        YR(7,1) - 0;        % lamGAMF
        H_tf - 1;           % Free final time condition
    ]);
return
```

analyzeResult.m

```
function [ ] = analyzeResult()
    % Define output path for plots
    outpath = '../../../../../Classwork/Spring 2014/AAE508/Project/Figures';

    tau = linspace(0,1,1000);
    % Load input file
    load('result_018LD.mat');

    const = out.const;
    scale = out.scale;

    R0 = scale(1); V0 = scale(2);

    h_fig1 = figure;
    h_fig2 = figure;
    cc=hsv(length(out.CONT));

    % Produce plots for continuation process
    for i=1:1:length(out.CONT)
        Z = deval(out.CONT(i).sol,tau);
        r = R0*Z(1,:)';
        theta = Z(2,:)';
        v = V0*Z(3,:)';
        gam = Z(4,:)';

        figure(1);

        plot(theta*const.re/1000,r/1000-const.re/1000,'Color',cc(i,:),'LineWidth',1);
        hold on; grid on;
        xlabel('Downrange (km)','FontSize',14);
        ylabel('Altitude (km)','FontSize',14);
        set(gca,'FontSize',14);

        figure(2);
        plot(v/1000,r/1000-const.re/1000,'Color',cc(i,:),'LineWidth',1);
        hold on; grid on;
        xlabel('Velocity (km/s)','FontSize',14);
        ylabel('Altitude (km)','FontSize',14);
        set(gca,'FontSize',14);
        axis square;
        legend(get(legend(gca(h_fig2)),'String'),strcat(['V_f = ',num2str(v(end)/1000,'%.1f'
            ),' km/s']));
    end
    set(legend(gca(h_fig1)),'Location','SouthEast','FontSize',14);
```

```
set(legend(gca(h_fig2)),'Location','SouthEastOutside','FontSize',14);
pos = get(h_fig2,'Position');
pos(3) = pos(3) + 100;
set(h_fig2,'Position',pos);

% Save the continuation plots
cur = pwd;
cd(outpath);
print(h_fig1,'-depsc','-r200','plot_cont_hS');
print(h_fig2,'-depsc','-r200','plot_cont_hV');
cd(cur);

% Load
Z = deval(out.CONT(end).sol,tau);
r = R0*Z(1,:)';
theta = Z(2,:)';
v = V0*Z(3,:)';
gam = Z(4,:)';
lamGam = Z(7,:)';
t = tau.*out.CONT(end).sol.parameters(1);

% Save the switching function plot
figure(1);
clf;
rho = const.rho0*exp(-(r-const.re)/const.h_scale);
H2 = rho.*lamGam.*const.LD.*v./(2*const.beta0);
plot(t,lamGam,'Color',[0 0.5 0]);
grid on;
xlabel('Time (s)','FontSize',14);
ylabel('\lambda_{\gamma}(t)','FontSize',14);
set(gca,'FontSize',14);
cd(outpath);
print(h_fig1,'-depsc','-r200','plot_lamgam');
cd(cur);

% Load the input files
sol_ld18 = out.CONT(end).sol;
Z_18 = deval(sol_ld18,tau);
load('result_ballistic.mat');
solb = out.CONT(end).sol;
Z_b = deval(solb,tau);

load('result_018LD_min.mat');
solmin = out.CONT(end).sol;
```

```
Z_min = deval(solmin,tau);

load('result_024LD.mat');
sol_ld24 = out.CONT(end).sol;
Z_24 = deval(sol_ld24,tau);

% Extract the required states and costates
r = R0*Z_18(1,:)';  r_b = R0*Z_b(1,:)';
theta = Z_18(2,:)'; theta_b = Z_b(2,:)';
v = V0*Z_18(3,:)';  v_b = V0*Z_b(3,:)';
gam = Z_18(4,:)';   gam_b = Z_b(4,:)';

r_min = R0*Z_min(1,:)';
theta_min = Z_min(2,:)';
v_min = V0*Z_min(3,:)';
gam_min = Z_min(4,:)';

r_24 = R0*Z_24(1,:)';
theta_24 = Z_24(2,:)';
v_24 = V0*Z_24(3,:)';
gam_24 = Z_24(4,:)';

lamGam = Z_18(7,:)';
lamGam_24 = Z_24(7,:)';

% Compute the time vectors
t = tau*sol_ld18.parameters(1);
t_b = tau*solb.parameters(1);
t_min = tau*solmin.parameters(1);
t_24 = tau*sol_ld24.parameters(1);

% Compute density for the 4 different trajectories
rho = const.rho0.*exp(-(r-const.re)./const.h_scale);
rho_b = const.rho0.*exp(-(r_b-const.re)./const.h_scale);
rho_24 = const.rho0.*exp(-(r_24-const.re)./const.h_scale);
rho_min = const.rho0.*exp(-(r_min-const.re)./const.h_scale);

Cl = const.LD*const.Cd;

% Calculate G-Loading for all 4 cases
gLoad = 0.5.*rho.*v.^2*const.Aref.*sqrt(Cl^2+const.Cd^2)/(const.mass*9.80665);
gLoad_b = 0.5.*rho_b.*v_b.^2*const.Aref.*sqrt(0^2+const.Cd^2)/(const.mass*9.80665);
gLoad_24 = 0.5.*rho_24.*v_24.^2*const.Aref.*sqrt(0.24^2+const.Cd^2)/(const.mass*9.80665)
    ;
```

```
gLoad_min = 0.5.*rho_min.*v_min.^2*const.Aref.*sqrt(Cl^2+const.Cd^2)/(const.mass
    *9.80665);

% Calculate integrated convective heat load for all 4 cases
qcdot = const.k*sqrt(rho./const.rn).*(v).^3;
qcdot_b = const.k*sqrt(rho_b./const.rn).*(v_b).^3;
qcdot_24 = const.k*sqrt(rho_24./const.rn).*(v_24).^3;
qcdot_min = const.k*sqrt(rho_min./const.rn).*(v_min).^3;

% Tauber sutton f(V) : Quadratic curve fit for parameters used in
% Tauber–Sutton radiative heating relation
p = [4.318831355255317e-05, -0.782410231351285, 3.539222367472642e+03];

% Compute radiative heating for  trajectory
C  = 4.736e4;
a  = 1.072e6*v.^(-1.88).*rho.^(-0.325);
b  = 1.22;
fV = p(1).*v.^2 + p(2).*v + p(3);
% Tauber and Sutton's radiatve heating equation
qrdot = C .* const.rn.^a .* rho.^b .* fV;

% Compute radiative heating for ballistic trajectory
C  = 4.736e4;
a  = 1.072e6*v_b.^(-1.88).*rho_b.^(-0.325);
b  = 1.22;
fV = p(1).*v_b.^2 + p(2).*v_b + p(3);
qrdot_b = C .* const.rn.^a .* rho_b.^b .* fV;

% Compute radiative heating for L/D=0.24 trajectory
C  = 4.736e4;
a  = 1.072e6*v_24.^(-1.88).*rho_24.^(-0.325);
b  = 1.22;
fV = p(1).*v_24.^2 + p(2).*v_24 + p(3);
qrdot_24 = C .* const.rn.^a .* rho_24.^b .* fV;

% Compute radiative heating for minimum-time trajectory
C  = 4.736e4;
a  = 1.072e6*v_min.^(-1.88).*rho_min.^(-0.325);
b  = 1.22;
fV = p(1).*v_min.^2 + p(2).*v_min + p(3);
qrdot_min = C .* const.rn.^a .* rho_min.^b .* fV;

% Calculate integrated heat load
q = trapz(t,qrdot);
```

```
fprintf('Integrated heat load (radiative) : %.2f J/cm^2\n',q);

set(0, 'DefaultFigureRenderer', 'Painters');

h1 = figure;
maxG1 = 5.0*ones(size(gLoad));
maxG2 = 9.0*ones(size(gLoad));
plot(t,gLoad,'Color',[0 0.5 0],'LineWidth',1);
hold on; grid on;
plot(t_b,gLoad_b,'b','LineWidth',1);
plot(t_min,gLoad_min,'Color',[0.5 0 0],'LineWidth',1);
plot(t,maxG1,'red--','LineWidth',3);
plot(t,maxG2,'red','LineWidth',3);

xlabel('Time (s)','FontSize',14);
ylabel('G Loading (G''s)','FontSize',14);
legend('Full Lift Down','Ballistic', 'Full Lift Up',...
        'Human tolerance (Typical)',...
        'Human tolerance (Peak)');
set(gca,'FontSize',14);
set(legend(gca),'FontSize',14);
ylim([0 13.5]);

h2 = figure;
plot(v/1000,r/1000-const.re/1000,'Color',[0 0.5 0],'LineWidth',1);
hold on; grid on;
plot(v_b/1000,r_b/1000-const.re/1000,'r','LineWidth',1);
plot(v_min/1000,r_min/1000-const.re/1000,'b','LineWidth',1);
xlabel('Velocity (km/s)','FontSize',14);
ylabel('Altitude (km)','FontSize',14);
set(gca,'FontSize',14);
legend('Full Lift Down','Ballistic','Full Lift Up','Location','NorthWest');
set(legend(gca),'FontSize',14);

if min(theta_b)<0
    theta_b = abs(theta_b);
end
h3 = figure;
plot(theta*const.re/1000,r/1000-const.re/1000,'Color',[0 0.5 0],'LineWidth',1);
hold on; grid on;
plot(theta_b*const.re/1000,r_b/1000-const.re/1000,'r','LineWidth',1);
plot(theta_min*const.re/1000,r_min/1000-const.re/1000,'b','LineWidth',1');
xlabel('Downrange (km)','FontSize',14);
ylabel('Altitude (km)','FontSize',14);
```

```
set(gca,'FontSize',14);
legend('Full Lift Down','Ballistic','Full Lift Up','Location','SouthEast');
set(legend(gca),'FontSize',14);
xlim([0 2600]);

h4 = figure;
plot(t,gam*180/pi,'Color',[0 0.5 0],'LineWidth',1);
hold on; grid on;
plot(t_b,gam_b*180/pi,'r','LineWidth',1);
plot(t_min,gam_min*180/pi,'b','LineWidth',1);

ylabel('Flight path angle (deg)','FontSize',14);
xlabel('Time (s)','FontSize',14);
legend('Full Lift Down','Ballistic','Full Lift Up','Location','SouthEast');
set(legend(gca),'FontSize',14);
set(gca,'FontSize',14);

h5 = figure;
maxqdot = 2000.0*ones(size(qrdot));
plot(t,qrdot+qcdot/10000,'Color',[0 0.5 0]);
hold on; grid on;
plot(t_b,qrdot_b+qcdot_b/10000,'b','LineWidth',1);
plot(t_min,qrdot_min+qcdot_min/10000,'Color',[0.5 0 0],'LineWidth',1);
plot(t,maxqdot,'red','LineWidth',4);
hold on; grid on;
xlabel('Time (s)','FontSize',14);
ylabel('Total heat rate (W/cm^2)','FontSize',14);
legend('Full Lift Down','Ballistic','Full Lift Up','PICA TPS','Location','SouthEast');
set(legend(gca),'FontSize',14);
set(gca,'FontSize',14);

    H2 = rho.*lamGam.*const.LD.*v./(2*const.beta0);
    u = zeros(size(H2));
    u(H2<0) = 1;
    u(H2>=0) = -1;

h6 = figure;
    plot(t,u,'LineWidth',1);
hold on; grid on;
ylim([-2 2]);
xlabel('Time (s)','FontSize',14);
ylabel('Control, u','FontSize',14);
set(gca,'FontSize',14);
```

```
h7 = figure;
maxqdot = 2000.0*ones(size(qrdot));
plot(t_24,qrdot_24+qcdot_24/10000,'Color',[0 0.5 0]);
hold on; grid on;
plot(t,qrdot+qcdot/10000,'b','LineWidth',1);
plot(t,maxqdot,'red','LineWidth',3);
hold on; grid on;
xlabel('Time (s)','FontSize',14);
ylabel('Total heat rate (W/cm^2)','FontSize',14);
lgnd = legend('L/D = 0.24','L/D = 0.18','PICA TPS','Location','SouthEast');
set(legend(gca),'FontSize',14);
set(lgnd,'FontSize',14);
ylim([0 2050]);

h8 = figure;
plot(theta_24*const.re/1000,r_24/1000-const.re/1000,'Color',[0 0.5 0],'LineWidth',1);
hold on; grid on;
plot(theta*const.re/1000,r/1000-const.re/1000,'b','LineWidth',1);
xlabel('Downrange (km)','FontSize',14);
ylabel('Altitude (km)','FontSize',14);
set(gca,'FontSize',14);
legend('L/D = 0.24','L/D = 0.18','Location','SouthEast');
set(legend(gca),'FontSize',14);

h9 = figure;
plot(t_24,gLoad_24,'Color',[0 0.5 0],'LineWidth',1);
hold on; grid on;
plot(t,gLoad,'b','LineWidth',1);
xlabel('Time (s)','FontSize',14);
ylabel('G Loading (G''s)','FontSize',14);
legend('L/D = 0.24','L/D = 0.18');
set(gca,'FontSize',14);
set(legend(gca),'FontSize',14);

h10 = figure;
plot(v_24/1000,r_24/1000-const.re/1000,'Color',[0 0.5 0],'LineWidth',1);
hold on; grid on;
plot(v/1000,r/1000-const.re/1000,'b','LineWidth',1);
xlabel('Velocity (km/s)','FontSize',14);
ylabel('Altitude (km)','FontSize',14);
set(gca,'FontSize',14);
legend('L/D = 0.24','L/D = 0.18','Location','NorthWest');
set(legend(gca),'FontSize',14);
```

```
    % Save the plots in high quality EPS format
    cur = pwd;
    cd(outpath);
    print(h1,'-r200','-depsc','plot_traj_G');
    print(h2,'-r200','-depsc','plot_traj_hV');
    print(h3,'-r200','-depsc','plot_traj');
    print(h4,'-r200','-depsc','plot_traj_gam');
    print(h5,'-r200','-depsc','plot_traj_heat');
    print(h6,'-r200','-depsc','plot_traj_u');

    print(h7,'-r200','-depsc','plot_traj24_heat');
    print(h8,'-r200','-depsc','plot_traj24_hS');
    print(h9,'-r200','-depsc','plot_traj24_G');
    print(h10,'-r200','-depsc','plot_traj24_hV');
    cd(cur);
end
```

References

[1]Lyne, J. E., Tauber, M. E., and Braun, R. D., "Parametric study of manned aerocapture Part I - Earth return from Mars," *Journal of Spacecraft and Rockets*, Vol. 29, No. 6, 1992, pp. 808–813.

[2]Vinh, N. X., Johnson, W., and Longuski, J., "Mars aerocapture using bank modulation," *Astrodynamics Specialist Conference*, American Institute of Aeronautics and Astronautics, 2000, pp. 497–506.

[3]Vinh, N. X., Busemann, A., and Culp, R. D., *Hypersonic and Planetary Entry Flight Mechanics*, The University of Michigan Press, Ann Arbor, 1980.

[4]Sutton, K. and Graves Jr., R. A., *A general stagnation-point convective heating equation for arbitrary gas mixtures*, NASA technical report; R-376, National Aeronautics and Space Administration, 1971.

[5]Tauber, M. E. and Sutton, K., "Stagnation-point radiative heating relations for Earth and Mars entries," *Journal of Spacecraft and Rockets*, Vol. 28, No. 1, 1991, pp. 40–42.

[6]Grant, M. J., Clark, I. G., and Braun, R. D., "Rapid design space exploration for conceptual design of hypersonic missions," *AIAA Atmospheric Flight Mechanics Conference and Exhibit, Portland, OR*, 2011, pp. 8–11.

[7]Trevino, L., "SpaceX Dragon Re-Entry Vehicle: Aerodynamics and Aerothermodynamics with Application to Base Heat-Shield Design," *6th International Planetary Probe Workshop*, Georgia Institute of Technology, 2008.

[8]Williams, D. R., "Earth Fact Sheet," `http://nssdc.gsfc.nasa.gov/planetary/factsheet/earthfact.html`, 2013, [Accessed: 2014-04-25].

[9]Bryson, A. E., *Applied optimal control: optimization, estimation and control*, CRC Press, 1975.

[10]Kikolaev, P., Stackpoole, M., Fan, W., Cruden, B., Waid, M., Moloney, P., Arepalli, S., Arnold, J., Partridge, H., and Yowell, L., "Carbon Nanotube-Enhanced Carbon-Phenenolic Ablator Material," *Materials Research Society Fall 2006 Meeting*, NASA Johnson Space Center, 2006, p. 15.

[11]Pandolf, K. B. and Burr, R. E., *Medical aspects of harsh environments*, Vol. 2, Government Printing Office, 2002.

[12]Grover, M., Sklyanskiy, E., Stelzner, A., and Sherwood, B., "Red Dragon-MSL Hybrid Landing Architecture for 2018," *LPI Contributions*, Vol. 1679, 2012, pp. 4216.

[13]Bonfiglio, E. P., Longuski, J. M., and Vinh, N. X., "Automated design of aerogravity-assist trajectories," *Journal of Spacecraft and Rockets*, Vol. 37, No. 6, 2000, pp. 768–775.