## <u>Review of "Survey of Numerical Methods for Trajectory Optimization" (Betts 1998)</u>

In this survey, Betts describes indirect methods that rely on finding the NOCs, and then solving a non-linear multipoint boundary value problem, as well as direct methods which do not require an analytic representation of the NOCs or initial guesses for the adjoint variables, but instead directly optimize the objective function using some parametric representation for the control variables. He stresses that both methods are similar in the sense that they use a Newton-based iteration to adjust a finite set of variables. Indirect methods have three main drawbacks – they need to solve complicated expressions for the NOCs, the region of convergence can be small, and the initial guess must be very good. Direct methods directly optimize the objective without the need for explicit NOCs and methods like direct shooting and direct transcription are widely used.

Broadly, talking about indirect and direct methods, Betts makes several interesting points. Firstly, in an indirect method, we often choose the absolute value of the gradient as a merit function, i.e. if the gradient absolute value decreases, we assume we're moving towards the local optimum. However, for a direct method, we need to be more careful and we take a step such that the objective function itself decreases. This means that the region of convergence for an indirect method is usually smaller than that for a direct method, i.e. the indirect method needs a better initial guess.

Secondly, to compute the gradient, indirect methods use analytic expressions, while direct methods use finite difference approximations. Choosing a forward difference versus central difference method to calculate the gradient information significantly impacts the accuracy of the gradient information (O(delta^2) vs O(delta)) but is also a tradeoff in terms of computational expense.

Thirdly, a big problem with indirect methods is deriving the necessary optimal conditions which involves solving differential equations, path constraints, and boundary conditions, which can be complicated to solve. In turn, this means that current indirect methods are not very flexible.

Betts describes direct shooting and indirect shooting methods in considerable detail.

Direct shooting uses a small number of NLP variables and rolls out the dynamics to achieve constraint satisfaction by solving an initial value problem. This is often used in problems where the number of NLP variables is low. In orbit transfer problems with impulsive burns, usually the thrust variation is not modeled very intricately because most real vehicles don't have the ability to implement a variable direction thrust even if one was computed. So direct shooting methods using four variables per burn, i.e. the time of ignition and the velocity increment, work perfectly well for problems like this. The performance of a direct shooting method worsens significantly as the number of NLP variables increases.

Indirect shooting methods differ from direct shooting methods because the control is defined at each time step. The biggest problem with these is the sensitivity to the initial guess. Some of Betts solutions to this problem include using the solution to a previously solved similar problem as a starting point for this new problem, assuming the solutions are in the same homotopy class. We can also use the sweep method to integrate state equations forward and adjoint equations backward. As with any indirect method, the NOCs need to be derived and solved explicitly.

When discussing implicit and explicit multi-step integration processes, Betts brings up the interesting point that the trajectory optimization problem is a boundary value problem, not an initial value problem, which means that any scheme is effectively implicit.

Another interesting point that Betts brings up is how in the aerospace field, the input data is often tabular in nature, be it because analytic physics-based representations are not possible, or that the data comes from experiments. This means that the numerical implementation of a trajectory optimization problem needs to be able to deal with tabular data. Linearly interpolating tabular data results in non-differentiability at the data points, leading to difficulties in numerical integration techniques that assume smooth differentiable functions. Betts suggests many alternatives for representing tabular data in a smooth functional form, e.g. using a quadratic drag polar or B-splines.

I didn't know about the methods of index reduction for path constraint functions if the partial derivative of the path constraint with respect to the control is not full rank. I was especially interested in how similar methods can be used if the Hessian with respect to u is a singular matrix, i.e. in the case of what Betts calls a "singular arc".

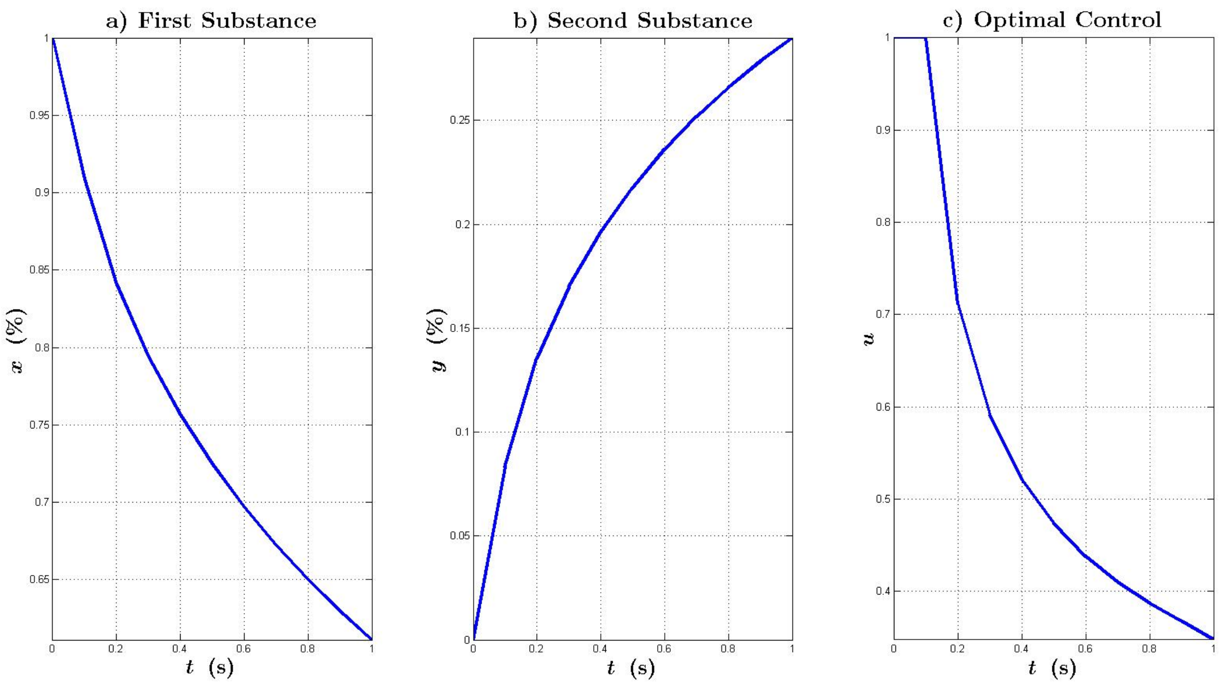
In summary, Betts also touches upon the main problems with dynamic programming, which is the curse of dimensionality, and genetic algorithms, which is the fact that discrete variables make it hard to determine convergence of a trajectory optimization algorithm and that they do not exploit gradient information.

Betts also hypothesizes that the biggest improvements to current methods will probably come from advances in high-index differential-algebraic equations, automatic differentiation, and sparse nonlinear programming.

## Problem 2

### Part a

Following is the plot for the time evolution of the states and control subject to the trapezoidal rule direct method. Code attached on following pages.



```matlab
% Problem (OCP)_1 from Pset 6

clear all; clf; clc; format long;

% Parameters and scenario
global N; N = 10;
global T; T = 1.;
global uMax; uMax = 1.0;
global x0; x0 = 1.;
global y0; y0 = 0.;

% Random initialization
uInit = 0.5*uMax*ones(N+1,1);
xInit = ones(N+1,1); yInit = zeros(N+1,1);
varInit = [xInit; yInit; uInit];

% Lower and upper bounds.
lb = zeros(3*N+3,1); ub = uMax*ones(3*N+3,1); % For the control: 0 \le u \le uMax
ub(1:N+1) = 1.; % For the state x : 0 \le x \le 1
ub(N+2:2*N+2) = 1.; % For the state y : 0 \le y \le 1

% Solving the problme via fmincon
options=optimoptions('fmincon','Display','iter','Algorithm','sqp','MaxFunEvals',100000,'MaxIter',10000);
% options=optimoptions('fmincon','Display','iter','Algorithm','sqp','MaxFunctionEvaluations',100000,'MaxIterations',10000);
[var,Fval,convergence] = fmincon(@cost,varInit,[],[],[],[],lb,ub,@constraint,options); % Solving the problem
convergence % = 1, good

% Collecting the solution. Note that var = [x;y;u]
x = var(1:N+1); y = var(N+2:2*N+2); u = var(2*N+3:3*N+3);
tState = zeros(N+1,1);
t = zeros(N+1,1);
for i = 1:N
    t(i+1) = t(i) + (1.0*T/(1.0*N));
end

% Plotting
fprintf('Optimal Final Quantity for the Second Substance = %f\n\n',Fval);
% subplot(131); plot(t,x,'linewidth',3);
% title('\textbf{a) First Substance}','interpreter','latex','FontSize',22,'FontWeight','bold');
% xlabel('\boldmath{$t$} \ \textbf{(s)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% ylabel('\boldmath{$x$} \ \textbf{(\%)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% xlim([-inf inf]);
% ylim([-inf inf]);
% grid on;
% subplot(132); plot(t,y,'linewidth',3) ;
% title('\textbf{b) Second Substance}','interpreter','latex','FontSize',22,'FontWeight','bold');
% xlabel('\boldmath{$t$} \ \textbf{(s)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% ylabel('\boldmath{$y$} \ \textbf{(\%)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% xlim([-inf inf]);
% ylim([-inf inf]);
% grid on;
% subplot(133); plot(t,u,'linewidth',3);
% title('\textbf{c) Optimal Control}','interpreter','latex','FontSize',22,'FontWeight','bold');
% xlabel('\boldmath{$t$} \ \textbf{(s)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% ylabel('\boldmath{$u$}','interpreter','latex','FontSize',20,'FontWeight','bold');
% xlim([-inf inf]);
% ylim([-inf inf]);
% grid on;
```

```
                                                          Norm of First-order
 Iter F-count            f(x) Feasibility  Steplength        step  optimality
    0      34   -0.000000e+00   5.000e-02                           1.000e+00
    1      68   -2.355430e-01   6.015e-03   1.000e+00   8.404e-01   2.996e-01
    2     102   -2.595025e-01   1.487e-04   1.000e+00   9.635e-02   2.447e-02
    3     136   -2.691728e-01   4.202e-04   1.000e+00   1.658e-01   2.300e-02
    4     171   -2.802969e-01   2.296e-04   1.000e+00   4.903e-01   1.343e-02
```

```
     5     205   -2.814745e-01   9.095e-05   1.000e+00   7.340e-02   1.337e-02
     6     239   -2.848175e-01   6.548e-04   1.000e+00   1.986e-01   1.322e-02
     7     273   -2.865277e-01   2.701e-04   1.000e+00   1.551e-01   1.946e-02
     8     307   -2.875543e-01   2.145e-04   1.000e+00   1.164e-01   9.424e-03
     9     341   -2.873440e-01   5.010e-05   1.000e+00   5.320e-02   7.636e-03
    10     375   -2.877465e-01   4.403e-05   1.000e+00   5.583e-02   7.914e-03
    11     409   -2.887661e-01   9.233e-05   1.000e+00   1.204e-01   1.138e-02
    12     443   -2.893493e-01   1.236e-04   1.000e+00   1.105e-01   1.082e-02
    13     477   -2.894912e-01   2.966e-05   1.000e+00   3.790e-02   7.160e-03
    14     511   -2.896184e-01   2.723e-05   1.000e+00   3.681e-02   3.220e-03
    15     545   -2.896203e-01   5.757e-06   1.000e+00   1.866e-02   2.536e-03
    16     579   -2.896686e-01   7.119e-06   1.000e+00   1.912e-02   2.495e-03
    17     613   -2.896967e-01   9.225e-06   1.000e+00   2.050e-02   1.997e-03
    18     647   -2.896969e-01   4.026e-06   1.000e+00   1.278e-02   1.219e-03
    19     681   -2.896988e-01   9.327e-07   1.000e+00   7.252e-03   1.153e-03
    20     715   -2.897080e-01   1.363e-06   1.000e+00   1.031e-02   1.593e-03
    21     749   -2.897188e-01   2.664e-06   1.000e+00   1.341e-02   1.973e-03
    22     783   -2.897193e-01   4.064e-07   1.000e+00   4.369e-03   1.452e-03
    23     817   -2.897202e-01   3.408e-07   1.000e+00   4.207e-03   2.091e-04
    24     851   -2.897196e-01   1.149e-08   1.000e+00   7.273e-04   1.729e-04
    25     885   -2.897203e-01   1.137e-07   1.000e+00   2.768e-03   3.421e-04
    26     919   -2.897203e-01   4.886e-08   1.000e+00   1.673e-03   2.749e-04
    27     953   -2.897203e-01   2.363e-08   1.000e+00   1.112e-03   1.060e-04
    28     987   -2.897203e-01   1.187e-09   1.000e+00   2.958e-04   3.901e-05
    29    1021   -2.897203e-01   9.043e-10   1.000e+00   2.835e-04   4.250e-05
    30    1055   -2.897203e-01   1.477e-09   1.000e+00   3.157e-04   4.239e-05

                                                          Norm of First-order
 Iter F-count            f(x) Feasibility  Steplength        step  optimality
    31    1089   -2.897203e-01   1.560e-09   1.000e+00   2.744e-04   3.140e-05
    32    1123   -2.897203e-01   4.151e-10   1.000e+00   1.342e-04   1.208e-05
    33    1157   -2.897203e-01   3.489e-11   1.000e+00   4.701e-05   6.886e-06
    34    1191   -2.897203e-01   3.684e-11   1.000e+00   4.258e-05   5.896e-06
    35    1225   -2.897203e-01   3.106e-11   1.000e+00   4.442e-05   5.937e-06
    36    1259   -2.897203e-01   1.542e-11   1.000e+00   3.203e-05   2.789e-06
    37    1293   -2.897203e-01   2.077e-12   1.000e+00   1.215e-05   9.103e-07

Local minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in 
feasible directions, to within the default value of the function tolerance,
and constraints are satisfied to within the default value of the constraint tolerance.




convergence =

     1

Optimal Final Quantity for the Second Substance = -0.289720
```

*Published with MATLAB® R2014a*

```
% Function providing equality and inequality constraints
% ceq(var) = 0 and c(var) \le 0

function [c,ceq] = constraint(var)

global N;
global T;

global x0;
global y0;

% Put here constraint inequalities
c = [];

% Note that var = [x;y;u]
x = var(1:N+1); y = var(N+2:2*N+2); u = var(2*N+3:3*N+3);

% Computing dynamical constraints via the trapezoidal rule
h = 1.0*T/(1.0*N);
for i = 1:N
    % Provide here dynamical constraints via the trapeziodal formula
    [xDyn_i,yDyn_i] = fDyn(x(i),y(i),u(i));
    [xDyn_ii,yDyn_ii] = fDyn(x(i+1),y(i+1),u(i+1));
    ceq(i) = x(i+1) - x(i) - h*(xDyn_i + xDyn_ii)/2;
    ceq(i+N) = y(i+1) - y(i) - h*(yDyn_i + yDyn_ii)/2;
end

% Put here initial conditions
ceq(1+2*N) = x(1) - 1;
ceq(2+2*N) = y(1) - 0;
```

```matlab
% Cost of the problem

function c = cost(var)

global N;

% Note that var = [x;y;u]
x = var(1:N+1); y = var(N+2:2*N+2); u = var(2*N+3:3*N+3);

% Put here the cost
c = -y(end);
```

```
% Dynamics of the problem

function [xDyn,yDyn] = fDyn(x,y,u)

% Put here the dynamics
xDyn = -x*u + y*u^2;
yDyn = x*u - 3*y*u^2;
```

*Published with MATLAB® R2014a*

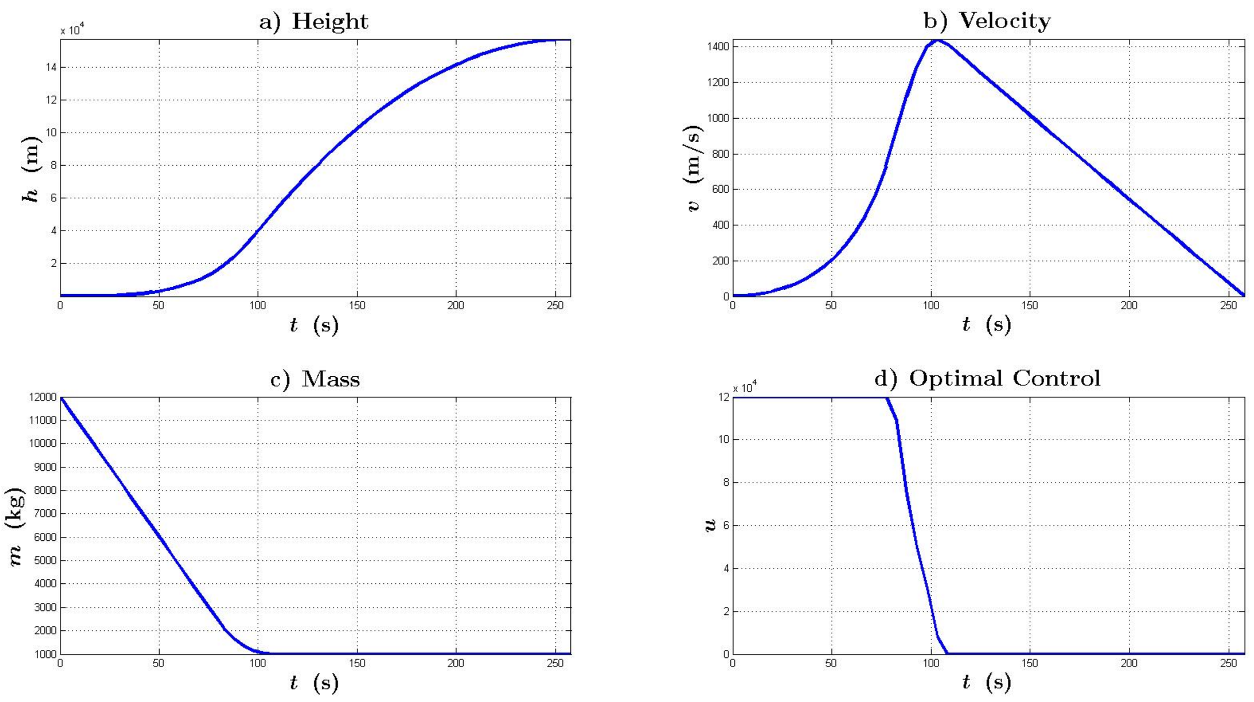
## Part b

The final cost at $t_f = 1$ is given by -0.289720 (see output of collocation function). Cost $= -y_f^*$. Therefore, $\boldsymbol{y_f^*} = \mathbf{28.97}\%$.

## Problem 3

### Part a

Following is the plot for the time evolution of the states and control subject to the trapezoidal rule direct method. Code attached on following pages.



```
% Problem (OCP)_2 from Pset 6 - Trapezoidal Rule

clear all; clf; clc; format long;

% Parameters
global N; N = 50; % Choose here the number of discretization points
global mu; mu = 3.9915e14;
global rE; rE = 6378145;
global h0; h0 = 7500;
global D; D = 5e-3;
global b; b = 1e-3;
global uMax; uMax = 1.2e5;

% Scenario
global T; T = 258.;
global y0; y0 = 0.;
global v0; v0 = 0.;
global m0; m0 = 12000;
global mf; mf = 1000;

% Bound on the state: better conditioning the formulation (see below)
global yMax; yMax = 5e6;
global vMax; vMax = 2000;

% Since this optimal control problem is highly nonlinear, without
% an appropriate intialization direct methods unlikely converge.
% In the following lines, we provide such initialization by recalling the
% solution that we obtained for the simplified Goddard problem in the Pset 5.
% For the height, we just select a stright-line in time connecting y0 to
% 1.5e5 (which is more or less the final height that we found in Pset 5).
% For the velocity, we select the average v(t) = vMax/2 in [0,tf].
% For the mass, we select a straight-line in time between 0 and tSw, the
% switching time computed in Pset 5 (see below).
% Finally, for the control, we select the maximal value u(t) = uMax in [0,tf].

% Finding what index NSw the time tSw corresponds to
global tSw; tSw = (m0 - mf)/(b*uMax);
h = (1.0*T/(1.0*N));
NSw = 0; indexFound = 0; iterator = 0;
while indexFound == 0
    % If iterator*h <= tSw < iteartor*h + h, then we have found the index
    if iterator*h <= tSw && tSw < (iterator + 1)*h
        NSw = iterator + 1;
        indexFound = 1;
    end
    iterator = iterator + 1;
end
uInit = zeros(N+1,1);
yInit = zeros(N+1,1);
vInit = 0.5*vMax*ones(N+1,1);
mInit = mf*ones(N+1,1);
% Initialization exxplained above
for i=1:N+1
    yInit(i) = y0*(1. - (i-1)*1.0/N) + 1.5e5*(i-1)*1.0/N;
    if (i-1) <= NSw
        mInit(i) = m0*(1. - (i-1)*1.0/NSw) + mf*(i-1)*1.0/NSw;
    end
    if i<= N
        if (i-1)*1.0*T/N < tSw
            uInit(i) = uMax;
        end
    end
end
% Initialization for fmincon
varInit = [yInit; vInit; mInit; uInit];

% Lower and upper bounds.
```

```
lb = zeros(4*N+4,1); ub = uMax*ones(4*N+4,1); % For the control: 0 \le u \le uMax
ub(1:N+1) = yMax; % For the state y : 0 \le y \le yMax
ub(N+2:2*N+2) = vMax; % For the state v : 0 \le v \le vMax
lb(2*N+3:3*N+3) = mf; ub(2*N+3:3*N+3) = m0; % For the state m : mf \le v \le m0

% Solving the problme via fmincon
options=optimoptions('fmincon','Display','iter','Algorithm','sqp','MaxFunEvals',100000,'MaxIter',10000);
% options=optimoptions('fmincon','Display','iter','Algorithm','sqp','MaxFunctionEvaluations',100000,'MaxIterations',10000);
[var,Fval,convergence] = fmincon(@cost,varInit,[],[],[],[],lb,ub,@constraint,options); % Solving the problem
convergence % = 1, good

% Collecting the solution. Note that var = [y;v;m;u]
y = var(1:N+1); v = var(N+2:2*N+2); m = var(2*N+3:3*N+3); u = var(3*N+4:4*N+4); % Collecting the solution
tState = zeros(N+1,1);
for i = 1:N
    tState(i+1) = tState(i) + (1.0*T/(1.0*N));
end
t = zeros(N+1,1);
for i = 1:N
    t(i+1) = t(i) + (1.0*T/(1.0*N));
end

% Plotting
% subplot(221); plot(tState,y,'linewidth',3);
% title('\textbf{a) Height}','interpreter','latex','FontSize',22,'FontWeight','bold');
% xlabel('\boldmath{$t$} \ \textbf{(s)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% ylabel('\boldmath{$h$} \ \textbf{(m)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% xlim([-inf inf]);
% ylim([-inf inf]);
% grid on;
% subplot(222); plot(tState,v,'linewidth',3) ;
% title('\textbf{b) Velocity}','interpreter','latex','FontSize',22,'FontWeight','bold');
% xlabel('\boldmath{$t$} \ \textbf{(s)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% ylabel('\boldmath{$v$} \ \textbf{(m/s)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% xlim([-inf inf]);
% ylim([-inf inf]);
% grid on;
% subplot(223); plot(tState,m,'linewidth',3) ;
% title('\textbf{c) Mass}','interpreter','latex','FontSize',22,'FontWeight','bold');
% xlabel('\boldmath{$t$} \ \textbf{(s)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% ylabel('\boldmath{$m$} \ \textbf{(kg)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% xlim([-inf inf]);
% ylim([-inf inf]);
% grid on;
% subplot(224); plot(t,u,'linewidth',3);
% title('\textbf{d) Optimal Control}','interpreter','latex','FontSize',22,'FontWeight','bold');
% xlabel('\boldmath{$t$} \ \textbf{(s)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% ylabel('\boldmath{$u$}','interpreter','latex','FontSize',20,'FontWeight','bold');
% xlim([-inf inf]);
% ylim([-inf inf]);
% grid on;
```

```
                                                          Norm of First-order
 Iter F-count            f(x) Feasibility  Steplength        step  optimality
    0     205   -1.500000e+05   2.160e+03                           1.000e+00
    1     410   -1.383047e+05   3.602e+00   1.000e+00   1.075e+05   6.461e+04
    2     617   -1.418327e+05   4.102e+00   7.000e-01   5.057e+04   6.822e+04
    3     822   -1.483241e+05   3.523e+00   1.000e+00   4.833e+04   9.815e+04
    4    1027   -1.500931e+05   5.709e-01   1.000e+00   1.475e+04   1.013e+05
    5    1232   -1.502196e+05   3.835e-03   1.000e+00   1.127e+03   1.014e+05
    6    1437   -1.502204e+05   1.564e-07   1.000e+00   7.180e+00   5.868e+02
    7    1642   -1.502212e+05   1.856e-07   1.000e+00   7.238e+00   2.042e+02
    8    1847   -1.502250e+05   4.626e-06   1.000e+00   3.619e+01   2.042e+02
    9    2052   -1.502441e+05   1.158e-04   1.000e+00   1.810e+02   2.028e+02
   10    2257   -1.503397e+05   2.912e-03   1.000e+00   9.065e+02   1.650e+02
   11    2462   -1.507327e+05   4.969e-02   1.000e+00   3.724e+03   1.650e+02
   12    2667   -1.507329e+05   5.195e-07   1.000e+00   3.809e+00   1.650e+02
   13    2872   -1.507331e+05   7.126e-08   1.000e+00   1.418e+00   1.650e+02
```

```
  14    3077   -1.507340e+05   2.389e-07   1.000e+00   5.171e+00   1.540e+02
  15    3282   -1.507346e+05   3.874e-07   1.000e+00   3.490e+00   1.169e+02
  16    3487   -1.507348e+05   6.257e-08   1.000e+00   1.416e+00   9.890e+01
  17    3692   -1.507349e+05   1.723e-09   1.000e+00   8.271e-01   8.842e+01
  18    3897   -1.507354e+05   4.257e-08   1.000e+00   4.100e+00   8.841e+01
  19    4102   -1.507357e+05   1.204e-08   1.000e+00   1.971e+00   7.893e+01
  20    4307   -1.507358e+05   9.076e-09   1.000e+00   1.576e+00   6.548e+01
  21    4512   -1.507364e+05   1.184e-07   1.000e+00   5.653e+00   6.349e+01
  22    4717   -1.507393e+05   2.959e-06   1.000e+00   2.827e+01   6.349e+01
  23    4922   -1.507538e+05   7.402e-05   1.000e+00   1.413e+02   6.348e+01
  24    5127   -1.508264e+05   1.857e-03   1.000e+00   7.072e+02   6.344e+01
  25    5332   -1.511901e+05   4.711e-02   1.000e+00   3.543e+03   5.901e+01
  26    5537   -1.522507e+05   4.113e-01   1.000e+00   1.029e+04   5.892e+01
  27    5742   -1.522599e+05   4.546e-06   1.000e+00   6.591e+01   5.893e+01
  28    5947   -1.522606e+05   1.423e-07   1.000e+00   5.303e+00   5.893e+01
  29    6152   -1.522620e+05   1.256e-07   1.000e+00   1.208e+01   5.893e+01
  30    6357   -1.522620e+05   4.735e-09   1.000e+00   3.541e-01   5.071e+01

                                                        Norm of First-order
 Iter F-count            f(x) Feasibility  Steplength       step  optimality
  31    6562   -1.522621e+05   3.479e-09   1.000e+00   1.029e+00   4.741e+01
  32    6767   -1.522623e+05   1.014e-08   1.000e+00   1.640e+00   3.837e+01
  33    6972   -1.522626e+05   4.522e-08   1.000e+00   3.428e+00   3.769e+01
  34    7177   -1.522642e+05   1.131e-06   1.000e+00   1.714e+01   3.769e+01
  35    7382   -1.522723e+05   2.830e-05   1.000e+00   8.570e+01   3.769e+01
  36    7587   -1.523127e+05   7.087e-04   1.000e+00   4.286e+02   3.768e+01
  37    7792   -1.525151e+05   1.787e-02   1.000e+00   2.145e+03   3.762e+01
  38    7997   -1.528313e+05   4.396e-02   1.000e+00   3.340e+03   3.752e+01
  39    8202   -1.528323e+05   2.736e-08   1.000e+00   7.553e+00   3.752e+01
  40    8407   -1.528324e+05   1.582e-09   1.000e+00   6.176e-01   3.140e+01
  41    8612   -1.528327e+05   3.925e-08   1.000e+00   3.088e+00   3.140e+01
  42    8817   -1.528343e+05   9.832e-07   1.000e+00   1.544e+01   3.140e+01
  43    9022   -1.528421e+05   2.460e-05   1.000e+00   7.721e+01   3.140e+01
  44    9227   -1.528809e+05   6.055e-04   1.000e+00   3.828e+02   3.139e+01
  45    9432   -1.528811e+05   1.684e-08   1.000e+00   1.984e+00   3.139e+01
  46    9637   -1.528820e+05   4.203e-07   1.000e+00   9.676e+00   3.139e+01
  47    9842   -1.528864e+05   1.051e-05   1.000e+00   4.838e+01   3.139e+01
  48   10047   -1.529083e+05   2.630e-04   1.000e+00   2.419e+02   3.138e+01
  49   10252   -1.530181e+05   6.610e-03   1.000e+00   1.210e+03   3.132e+01
  50   10457   -1.535694e+05   1.696e-01   1.000e+00   6.069e+03   3.104e+01
  51   10662   -1.539207e+05   6.877e-02   1.000e+00   3.811e+03   3.087e+01
  52   10867   -1.539221e+05   6.668e-08   1.000e+00   1.013e+01   3.087e+01
  53   11072   -1.539221e+05   2.402e-10   1.000e+00   2.677e-01   3.087e+01
  54   11277   -1.539222e+05   3.561e-09   1.000e+00   1.339e+00   3.087e+01
  55   11482   -1.539228e+05   8.981e-08   1.000e+00   6.693e+00   3.087e+01
  56   11687   -1.539239e+05   3.120e-07   1.000e+00   1.227e+01   3.087e+01
  57   11892   -1.539239e+05   1.071e-09   1.000e+00   5.357e-01   3.087e+01
  58   12097   -1.539241e+05   2.656e-08   1.000e+00   2.679e+00   3.087e+01
  59   12302   -1.539250e+05   6.632e-07   1.000e+00   1.339e+01   3.087e+01
  60   12507   -1.539295e+05   1.658e-05   1.000e+00   6.696e+01   3.086e+01

                                                        Norm of First-order
 Iter F-count            f(x) Feasibility  Steplength       step  optimality
  61   12712   -1.539521e+05   4.151e-04   1.000e+00   3.348e+02   3.084e+01
  62   12917   -1.540654e+05   1.044e-02   1.000e+00   1.675e+03   3.071e+01
  63   13122   -1.546322e+05   2.676e-01   1.000e+00   8.370e+03   3.006e+01
  64   13327   -1.555254e+05   6.932e-01   1.000e+00   1.305e+04   2.899e+01
  65   13532   -1.555348e+05   1.843e-06   1.000e+00   8.475e+01   2.900e+01
  66   13737   -1.555349e+05   1.237e-07   1.000e+00   9.483e-01   2.900e+01
  67   13942   -1.555354e+05   4.536e-07   1.000e+00   3.057e+00   2.900e+01
  68   14147   -1.555355e+05   4.119e-08   1.000e+00   9.729e-01   2.900e+01
  69   14352   -1.555357e+05   1.850e-07   1.000e+00   1.643e+00   2.335e+01
  70   14557   -1.555358e+05   2.874e-09   1.000e+00   7.418e-01   1.829e+01
  71   14762   -1.555361e+05   6.435e-08   1.000e+00   3.709e+00   1.828e+01
  72   14967   -1.555377e+05   1.638e-06   1.000e+00   1.855e+01   1.828e+01
  73   15172   -1.555456e+05   4.105e-05   1.000e+00   9.273e+01   1.823e+01
  74   15377   -1.555510e+05   1.902e-05   1.000e+00   6.303e+01   1.820e+01
  75   15582   -1.555546e+05   2.011e-05   1.000e+00   2.248e+02   1.820e+01
  76   15787   -1.555727e+05   5.030e-04   1.000e+00   1.124e+03   1.820e+01
  77   15992   -1.556376e+05   6.578e-03   1.000e+00   4.055e+03   1.820e+01
  78   16197   -1.556374e+05   1.322e-09   1.000e+00   6.623e-01   1.820e+01
  79   16402   -1.556374e+05   8.549e-11   1.000e+00   1.407e-01   1.820e+01
  80   16607   -1.556375e+05   1.488e-09   1.000e+00   7.035e-01   1.820e+01
  81   16812   -1.556377e+05   3.724e-08   1.000e+00   3.518e+00   1.820e+01
  82   17017   -1.556388e+05   9.350e-07   1.000e+00   1.759e+01   1.820e+01
  83   17222   -1.556442e+05   2.337e-05   1.000e+00   8.794e+01   1.820e+01
  84   17427   -1.556712e+05   5.850e-04   1.000e+00   4.397e+02   1.820e+01
  85   17632   -1.558062e+05   1.472e-02   1.000e+00   2.199e+03   1.820e+01
  86   17837   -1.560135e+05   3.516e-02   1.000e+00   3.374e+03   1.820e+01
  87   18042   -1.560136e+05   9.209e-09   1.000e+00   3.639e+00   1.820e+01
  88   18247   -1.560137e+05   1.456e-10   1.000e+00   2.128e-01   1.820e+01
  89   18452   -1.560137e+05   3.912e-09   1.000e+00   1.064e+00   1.820e+01
  90   18657   -1.560140e+05   9.877e-08   1.000e+00   5.319e+00   1.820e+01

                                                              Norm of First-order
 Iter F-count            f(x) Feasibility  Steplength            step  optimality
  91   18862   -1.560154e+05   2.471e-06   1.000e+00   2.660e+01   1.820e+01
  92   19067   -1.560223e+05   6.179e-05   1.000e+00   1.330e+02   1.820e+01
  93   19272   -1.560572e+05   1.548e-03   1.000e+00   6.649e+02   1.820e+01
  94   19477   -1.562312e+05   3.907e-02   1.000e+00   3.324e+03   1.820e+01
  95   19682   -1.570949e+05   1.011e+00   1.000e+00   1.650e+04   1.820e+01
  96   19887   -1.569294e+05   3.987e-02   1.000e+00   3.160e+03   1.820e+01
  97   20092   -1.569213e+05   9.737e-05   1.000e+00   1.574e+02   1.779e+01
  98   20297   -1.569213e+05   6.041e-10   1.000e+00   3.814e-01   1.780e+01
  99   20502   -1.569213e+05   1.145e-10   1.000e+00   2.031e-01   1.779e+01
 100   20707   -1.569213e+05   2.918e-09   1.000e+00   1.015e+00   1.779e+01
 101   20912   -1.569215e+05   7.216e-08   1.000e+00   5.076e+00   1.778e+01
 102   21117   -1.569222e+05   1.807e-06   1.000e+00   2.537e+01   1.770e+01
 103   21322   -1.569260e+05   4.493e-05   1.000e+00   1.266e+02   1.734e+01
 104   21527   -1.569444e+05   1.096e-03   1.000e+00   6.254e+02   1.553e+01
 105   21732   -1.570279e+05   2.425e-02   1.000e+00   2.955e+03   1.022e+01
 106   21937   -1.572442e+05   2.409e-01   1.000e+00   9.812e+03   2.595e+01
 107   22142   -1.571377e+05   1.251e-02   1.000e+00   1.920e+03   2.859e+01
 108   22347   -1.571363e+05   5.459e-04   1.000e+00   7.539e+02   1.567e+01
 109   22552   -1.571358e+05   6.095e-05   1.000e+00   2.531e+02   1.515e+00
 110   22757   -1.571357e+05   1.390e-08   1.000e+00   2.179e+00   8.903e-01
 111   22962   -1.571357e+05   2.620e-09   1.000e+00   1.592e+00   8.602e-01
 112   23167   -1.571357e+05   1.912e-08   1.000e+00   4.278e+00   8.508e-01
 113   23372   -1.571357e+05   5.591e-08   1.000e+00   7.316e+00   8.346e-01
 114   23577   -1.571357e+05   9.721e-08   1.000e+00   9.637e+00   8.131e-01
 115   23782   -1.571357e+05   3.434e-07   1.000e+00   1.809e+01   7.719e-01
 116   23987   -1.571357e+05   7.926e-07   1.000e+00   2.754e+01   7.173e-01
 117   24192   -1.571357e+05   2.168e-06   1.000e+00   4.574e+01   8.610e-01
 118   24397   -1.571358e+05   5.312e-06   1.000e+00   7.271e+01   1.084e+00
 119   24602   -1.571358e+05   1.320e-05   1.000e+00   1.199e+02   1.432e+00
 120   24807   -1.571360e+05   2.628e-05   1.000e+00   1.925e+02   1.908e+00

                                                              Norm of First-order
 Iter F-count            f(x) Feasibility  Steplength            step  optimality
 121   25012   -1.571365e+05   3.411e-05   1.000e+00   3.082e+02   2.442e+00
 122   25217   -1.571372e+05   1.147e-04   1.000e+00   4.507e+02   2.603e+00
 123   25422   -1.571379e+05   2.233e-04   1.000e+00   5.254e+02   2.255e+00
 124   25627   -1.571381e+05   1.041e-04   1.000e+00   3.927e+02   1.217e+00
 125   25832   -1.571380e+05   3.013e-05   1.000e+00   1.725e+02   2.977e-01
 126   26037   -1.571380e+05   5.790e-07   1.000e+00   4.400e+01   7.066e-02
 127   26242   -1.571380e+05   1.965e-08   1.000e+00   5.319e+00   4.792e-02
 128   26447   -1.571380e+05   9.919e-11   1.000e+00   4.016e-01   4.759e-02
 129   26652   -1.571380e+05   8.151e-11   1.000e+00   3.166e-01   4.723e-02
 130   26857   -1.571380e+05   3.601e-10   1.000e+00   7.526e-01   4.687e-02
 131   27062   -1.571380e+05   1.022e-09   1.000e+00   1.121e+00   4.555e-02
 132   27267   -1.571380e+05   2.996e-09   1.000e+00   1.989e+00   4.343e-02
 133   27472   -1.571380e+05   4.901e-09   1.000e+00   2.394e+00   4.024e-02
 134   27677   -1.571380e+05   2.455e-08   1.000e+00   5.542e+00   5.409e-02
 135   27882   -1.571380e+05   3.952e-08   1.000e+00   7.083e+00   7.049e-02
 136   28087   -1.571380e+05   1.242e-07   1.000e+00   1.227e+01   1.210e-01
 137   28292   -1.571380e+05   3.776e-07   1.000e+00   1.999e+01   2.099e-01
 138   28497   -1.571380e+05   8.627e-07   1.000e+00   3.034e+01   3.407e-01
 139   28702   -1.571380e+05   1.974e-06   1.000e+00   4.825e+01   5.104e-01
```

```
   140   28907   -1.571380e+05   5.414e-06   1.000e+00   7.498e+01   6.285e-01
   141   29112   -1.571381e+05   1.520e-05   1.000e+00   9.405e+01   5.331e-01
   142   29317   -1.571381e+05   8.756e-06   1.000e+00   7.580e+01   3.720e-01
   143   29522   -1.571381e+05   7.624e-07   1.000e+00   3.068e+01   1.273e-01
   144   29727   -1.571381e+05   4.451e-08   1.000e+00   7.298e+00   1.982e-02
   145   29932   -1.571381e+05   1.324e-09   1.000e+00   9.539e-01   2.543e-03
   146   30137   -1.571381e+05   2.319e-11   1.000e+00   7.672e-02   2.442e-03
   147   30139   -1.571381e+05   2.319e-11   4.900e-01   1.615e-02   1.018e-07

Local minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the default value of the function tolerance,
and constraints are satisfied to within the default value of the constraint tolerance.




convergence =

     1
```

```
% Function providing equality and inequality constraints
% ceq(var) = 0 and c(var) \le 0

function [c,ceq] = constraint(var)

global N;
global T;

global y0;
global v0;
global m0;
global mf;

% Put here constraint inequalities
c = [];

% Note that var = [y;v;m;u]
y = var(1:N+1); v = var(N+2:2*N+2); m = var(2*N+3:3*N+3); u = var(3*N+4:4*N+4); % Note: var = [y;v;m;u]

% Computing dynamical constraints via the trapezoidal rule
h = 1.0*T/(1.0*N);
for i = 1:N
    % Provide here dynamical constraints via the trapeziodal formula
    [yDyn_i,vDyn_i,mDyn_i] = fDyn(y(i),v(i),m(i),u(i));
    [yDyn_ii,vDyn_ii,mDyn_ii] = fDyn(y(i+1),v(i+1),m(i+1),u(i+1));
    ceq(i) = y(i+1) - y(i) - h*(yDyn_i + yDyn_ii)/2;
    ceq(i+N) = v(i+1) - v(i) - h*(vDyn_i + vDyn_ii)/2;
    ceq(i+2*N) = m(i+1) - m(i) - h*(mDyn_i + mDyn_ii)/2;
end

% Put here initial and final conditions
ceq(1+3*N) = y(1) - y0;
ceq(2+3*N) = v(1) - v0;
ceq(3+3*N) = m(1) - m0;
ceq(4+3*N) = m(end) - mf;
```

```matlab
% Cost of the problem

function c = cost(var)

global N;

% Note that var = [y;v;m;u]
y = var(1:N+1); v = var(N+2:2*N+2); m = var(2*N+3:3*N+3); u = var(3*N+4:4*N+3);

% Put here the cost
c = -y(end);
```

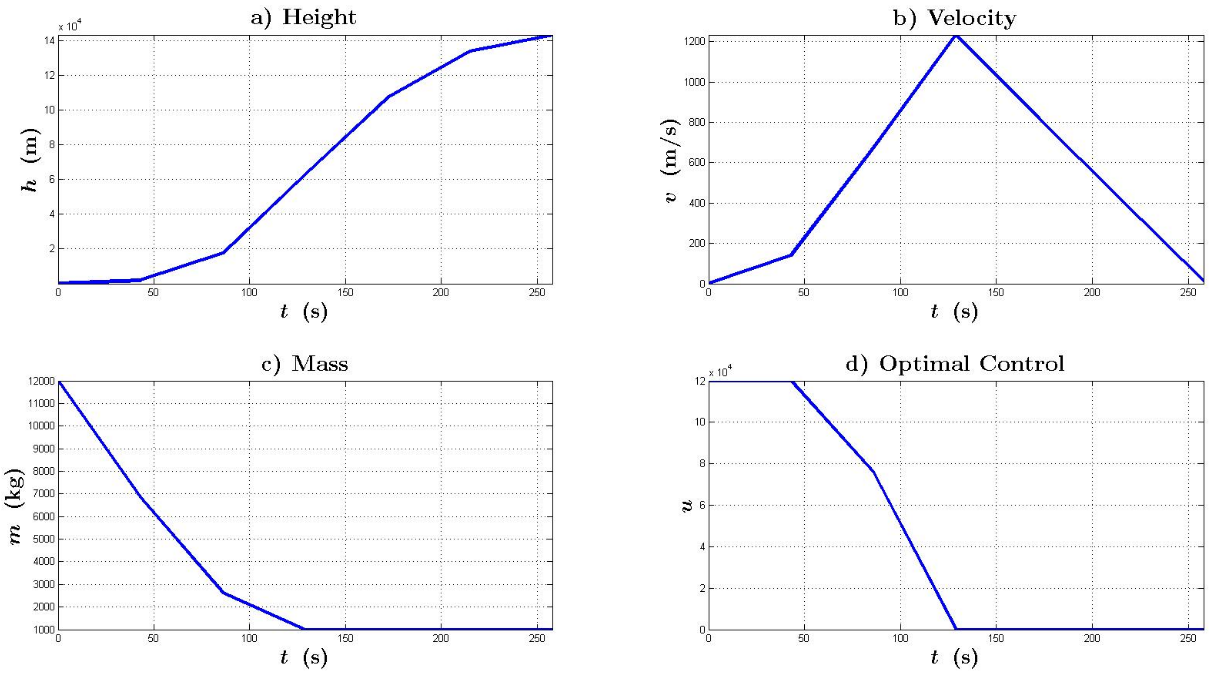
```
% Dynamics of the problem

function [yDyn,vDyn,mDyn] = fDyn(y,v,m,u)

global D;
global b;

g = gFunc(y);
rho = normRhoFunc(y);

% Put here the dynamics
yDyn = v;
vDyn = u/m - g - (D/m)*rho*v^2;
mDyn = -b*u;
```

## Part b

The minimum number of discretization points N required to get the MATLAB function using trapezoidal rule direct method to converge (with exitflag = 1) was **50**.

## Part c

Following is the plot for the time evolution of the states and control subject to the Hermite-Simpson rule direct method.



The “cost” and “fdyn” functions remain unchanged. The code for the “collocation” and “constraint” functions are attached on following pages.

```
% Problem (OCP)_2 from Pset 6 - Hermite-Simpson Rule

clear all; clf; clc; format long;

% Parameters
global N; N = 6; % Choose here the number of discretization points
global mu; mu = 3.9915e14;
global rE; rE = 6378145;
global h0; h0 = 7500;
global D; D = 5e-3;
global b; b = 1e-3;
global uMax; uMax = 1.2e5;

% Scenario
global T; T = 258.;
global y0; y0 = 0.;
global v0; v0 = 0.;
global m0; m0 = 12000;
global mf; mf = 1000;

% Bound on the state: better conditioning the formulation (see below)
global yMax; yMax = 5e6;
global vMax; vMax = 2000;

% Since this optimal control problem is highly nonlinear, without
% an appropriate intialization direct methods unlikely converge.
% In the following lines, we provide such initialization by recalling the
% solution that we obtained for the simplified Goddard problem in the Pset 5.
% For the height, we just select a stright-line in time connecting y0 to
% 1.5e5 (which is more or less the final height that we found in Pset 5).
% For the velocity, we select the average v(t) = vMax/2 in [0,tf].
% For the mass, we select a straight-line in time between 0 and tSw, the
% switching time computed in Pset 5 (see below).
% Finally, for the control, we select the maximal value u(t) = uMax in [0,tf].

% Finding what index NSw the time tSw corresponds to
global tSw; tSw = (m0 - mf)/(b*uMax);
h = (1.0*T/(1.0*N));
NSw = 0; indexFound = 0; iterator = 0;
while indexFound == 0
    % If iterator*h <= tSw < itеartor*h + h, then we have found the index
    if iterator*h <= tSw && tSw < (iterator + 1)*h
        NSw = iterator + 1;
        indexFound = 1;
    end
    iterator = iterator + 1;
end
uInit = zeros(N+1,1);
yInit = zeros(N+1,1);
vInit = 0.5*vMax*ones(N+1,1);
mInit = mf*ones(N+1,1);
% Initialization exxplained above
for i=1:N+1
    yInit(i) = y0*(1. - (i-1)*1.0/N) + 1.5e5*(i-1)*1.0/N;
    if (i-1) <= NSw
        mInit(i) = m0*(1. - (i-1)*1.0/NSw) + mf*(i-1)*1.0/NSw;
    end
    if i<= N
        if (i-1)*1.0*T/N < tSw
            uInit(i) = uMax;
        end
    end
end
% Initialization for fmincon
varInit = [yInit; vInit; mInit; uInit];

% Lower and upper bounds.
```

```
lb = zeros(4*N+4,1); ub = uMax*ones(4*N+4,1); % For the control: 0 \le u \le uMax
ub(1:N+1) = yMax; % For the state y : 0 \le y \le yMax
ub(N+2:2*N+2) = vMax; % For the state v : 0 \le v \le vMax
lb(2*N+3:3*N+3) = mf; ub(2*N+3:3*N+3) = m0; % For the state m : mf \le v \le m0

% Solving the problme via fmincon
options=optimoptions('fmincon','Display','iter','Algorithm','sqp','MaxFunEvals',100000,'MaxIter',10000);
% options=optimoptions('fmincon','Display','iter','Algorithm','sqp','MaxFunctionEvaluations',100000,'MaxIterations',10000);
[var,Fval,convergence] = fmincon(@cost,varInit,[],[],[],[],lb,ub,@constraint,options); % Solving the problem
convergence % = 1, good

% Collecting the solution. Note that var = [y;v;m;u]
y = var(1:N+1); v = var(N+2:2*N+2); m = var(2*N+3:3*N+3); u = var(3*N+4:4*N+4); % Collecting the solution
tState = zeros(N+1,1);
for i = 1:N
    tState(i+1) = tState(i) + (1.0*T/(1.0*N));
end
t = zeros(N+1,1);
for i = 1:N
    t(i+1) = t(i) + (1.0*T/(1.0*N));
end

% Plotting
% subplot(221); plot(tState,y,'linewidth',3);
% title('\textbf{a) Height}','interpreter','latex','FontSize',22,'FontWeight','bold');
% xlabel('\boldmath{$t$} \ \textbf{(s)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% ylabel('\boldmath{$h$} \ \textbf{(m)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% xlim([-inf inf]);
% ylim([-inf inf]);
% grid on;
% subplot(222); plot(tState,v,'linewidth',3) ;
% title('\textbf{b) Velocity}','interpreter','latex','FontSize',22,'FontWeight','bold');
% xlabel('\boldmath{$t$} \ \textbf{(s)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% ylabel('\boldmath{$v$} \ \textbf{(m/s)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% xlim([-inf inf]);
% ylim([-inf inf]);
% grid on;
% subplot(223); plot(tState,m,'linewidth',3) ;
% title('\textbf{c) Mass}','interpreter','latex','FontSize',22,'FontWeight','bold');
% xlabel('\boldmath{$t$} \ \textbf{(s)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% ylabel('\boldmath{$m$} \ \textbf{(kg)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% xlim([-inf inf]);
% ylim([-inf inf]);
% grid on;
% subplot(224); plot(t,u,'linewidth',3);
% title('\textbf{d) Optimal Control}','interpreter','latex','FontSize',22,'FontWeight','bold');
% xlabel('\boldmath{$t$} \ \textbf{(s)}','interpreter','latex','FontSize',20,'FontWeight','bold');
% ylabel('\boldmath{$u$}','interpreter','latex','FontSize',20,'FontWeight','bold');
% xlim([-inf inf]);
% ylim([-inf inf]);
% grid on;
```

```
                                                          Norm of First-order
 Iter F-count            f(x) Feasibility  Steplength        step  optimality
    0      29   -1.500000e+05   2.195e+04                           1.000e+00
    1      58   -1.300265e+05   1.408e+02   1.000e+00   6.641e+04   1.483e+06
    2      87   -1.302402e+05   5.410e-01   1.000e+00   1.256e+03   1.517e+04
    3     116   -1.302413e+05   1.637e-05   1.000e+00   3.926e+00   2.542e+01
    4     145   -1.302419e+05   3.305e-07   1.000e+00   8.138e-01   4.686e+00
    5     174   -1.302449e+05   6.053e-06   1.000e+00   4.069e+00   4.686e+00
    6     203   -1.302600e+05   1.458e-04   1.000e+00   2.034e+01   4.686e+00
    7     232   -1.303357e+05   3.643e-03   1.000e+00   1.017e+02   4.686e+00
    8     261   -1.305159e+05   2.088e-02   1.000e+00   2.422e+02   4.685e+00
    9     290   -1.305188e+05   1.050e-05   1.000e+00   4.154e+00   4.685e+00
   10     319   -1.305325e+05   2.386e-04   1.000e+00   2.021e+01   4.685e+00
   11     348   -1.306013e+05   6.002e-03   1.000e+00   1.011e+02   4.685e+00
   12     377   -1.309454e+05   1.512e-01   1.000e+00   5.056e+02   4.685e+00
   13     406   -1.310556e+05   1.555e-02   1.000e+00   1.618e+02   4.685e+00
```

```
    14     435   -1.310562e+05   3.920e-07   1.000e+00   1.055e+00   4.685e+00  
    15     464   -1.310589e+05   1.172e-05   1.000e+00   4.982e+00   4.686e+00  
    16     493   -1.310720e+05   2.956e-04   1.000e+00   2.491e+01   4.686e+00  
    17     522   -1.311378e+05   7.465e-03   1.000e+00   1.246e+02   4.691e+00  
    18     551   -1.314671e+05   1.878e-01   1.000e+00   6.232e+02   4.714e+00  
    19     580   -1.324321e+05   1.631e+00   1.000e+00   1.825e+03   4.781e+00  
    20     609   -1.324546e+05   6.150e-04   1.000e+00   4.081e+01   4.782e+00  
    21     638   -1.325160e+05   7.243e-03   1.000e+00   1.159e+02   4.787e+00  
    22     667   -1.328238e+05   1.825e-01   1.000e+00   5.813e+02   4.808e+00  
    23     696   -1.338126e+05   1.903e+00   1.000e+00   1.866e+03   4.879e+00  
    24     725   -1.338182e+05   7.383e-08   1.000e+00   9.419e+00   4.865e+00  
    25     754   -1.338186e+05   1.038e-07   1.000e+00   8.835e-01   4.865e+00  
    26     783   -1.338207e+05   2.724e-06   1.000e+00   4.418e+00   4.867e+00  
    27     812   -1.338308e+05   6.745e-05   1.000e+00   2.209e+01   4.876e+00  
    28     841   -1.338816e+05   1.686e-03   1.000e+00   1.105e+02   4.921e+00  
    29     870   -1.341357e+05   4.229e-02   1.000e+00   5.527e+02   5.148e+00  
    30     899   -1.354107e+05   1.075e+00   1.000e+00   2.771e+03   6.304e+00  

                                                            Norm of First-order
  Iter F-count            f(x) Feasibility  Steplength        step  optimality
    31     928   -1.413389e+05   2.438e+01   1.000e+00   1.283e+04   1.211e+01  
    32     957   -1.420058e+05   1.624e-01   1.000e+00   1.316e+03   1.260e+01  
    33     986   -1.420188e+05   4.979e-05   1.000e+00   3.210e+01   1.261e+01  
    34    1015   -1.420192e+05   2.789e-07   1.000e+00   8.408e-01   1.261e+01  
    35    1044   -1.420213e+05   7.602e-06   1.000e+00   4.199e+00   1.261e+01  
    36    1073   -1.420319e+05   1.884e-04   1.000e+00   2.100e+01   1.263e+01  
    37    1102   -1.420849e+05   4.593e-03   1.000e+00   1.050e+02   1.271e+01  
    38    1131   -1.423506e+05   1.157e-01   1.000e+00   5.262e+02   1.313e+01  
    39    1160   -1.432112e+05   1.219e+00   1.000e+00   1.701e+03   5.343e+00  
    40    1189   -1.432192e+05   1.178e-07   1.000e+00   1.221e+01   5.042e+00  
    41    1218   -1.432192e+05   1.819e-11   1.000e+00   8.017e-07   5.684e-14  

Local minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in 
feasible directions, to within the default value of the function tolerance,
and constraints are satisfied to within the default value of the constraint tolerance.




convergence =

     1
```

```
% Function providing equality and inequality constraints
% ceq(var) = 0 and c(var) \le 0

function [c,ceq] = constraint(var)

global N;
global T;

global y0;
global v0;
global m0;
global mf;

% Put here constraint inequalities
c = [];

% Note that var = [y;v;m;u]
y = var(1:N+1); v = var(N+2:2*N+2); m = var(2*N+3:3*N+3); u = var(3*N+4:4*N+4); % Note: var = [y;v;m;u]

% Computing dynamical constraints via the Hermite-Simpson rule
h = 1.0*T/(1.0*N);
for i = 1:N
    % Provide here dynamical constraints via the Hermite-Simpson formula
    [yDyn_i,vDyn_i,mDyn_i] = fDyn(y(i),v(i),m(i),u(i));
    [yDyn_ii,vDyn_ii,mDyn_ii] = fDyn(y(i+1),v(i+1),m(i+1),u(i+1));

    y_ic = (1./2.)*(y(i) + y(i+1)) + (1.0*T/(1.0*N))/8.*(yDyn_i - yDyn_ii); % Evaluating state and control at collocation points via the Hermite-Simpson formula
    v_ic = (1./2.)*(v(i) + v(i+1)) + (1.0*T/(1.0*N))/8.*(vDyn_i - vDyn_ii);
    m_ic = (1./2.)*(m(i) + m(i+1)) + (1.0*T/(1.0*N))/8.*(mDyn_i - mDyn_ii);
    u_ic = (u(i) + u(i+1))/2.;

    [yDyn_ic,vDyn_ic,mDyn_ic] = fDyn(y_ic,v_ic,m_ic,u_ic); % Evaluating dynamics at collocation points

    ceq(i) = y(i+1) - y(i) - ((1.0*T/(1.0*N))/6.0)*(yDyn_i + 4*yDyn_ic + yDyn_ii);
    ceq(i+N) = v(i+1) - v(i) - ((1.0*T/(1.0*N))/6.0)*(vDyn_i + 4*vDyn_ic + vDyn_ii);;
    ceq(i+2*N) = m(i+1) - m(i) - ((1.0*T/(1.0*N))/6.0)*(mDyn_i + 4*mDyn_ic + mDyn_ii);;
end

% Put here initial and final conditions
ceq(1+3*N) = y(1) - y0;
ceq(2+3*N) = v(1) - v0;
ceq(3+3*N) = m(1) - m0;
ceq(4+3*N) = m(end) - mf;
```

## Part d

The minimum number of discretization points N required to get the MATLAB function using Hermite-Simpson rule direct method to converge (with exitflag = 1) was **6**.