```
% 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;
```

Published with MATLAB® R2014a