Table of Contents


```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
% This is the driver script for the fourth question in homework 3
%
% Created by: Johnathan Tucker
%
% Date Created: 9/17/2019
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%
```

Housekeeping

```
clear all;
close all;
clc;
```

Create the necessary constants

First get t_0 from r_1_norm

```
a = 7735.6; % [km]
theta = 4.4588;
e = 0.6271;
T = 28801; % [s]
r_1_norm = 5569.5;% [km]
E_1 = -2*atan(sqrt((1-e)/(1+e))*tan(theta/2));
Me_1 = E_1 - e*sin(E_1);
t_0 = Me_1 * T/(2*pi);
t_0 = T-t_0;
% Generally this problems are begin with a known time and period
t = t_0 + 7200; % [s]

% And if position is to be calculated then eccentricity and semi-major
 axis
% will be needed

mu = 22030;
h = 10168;
% Calculate the mean anomaly
M_e = (t/T) * 2*pi; % [rad]
```

Begin implementing Newton-Raphson

This error was chosen based off numerical analysis standards. This is how many digits MATLAB is precise to and therefore it's our tolerance.

```
tolerance = 1*(10^(-16));

% As per the book on page 153 choose an initial guess for E
if M_e < pi
E = M_e + e/2;
else
E = M_e - e/2;
end
% This initial value of delta is necessary for the while loop to start
delta = 1;
% Every iteration check if the difference is within the tolerance
while abs(delta) > tolerance
    delta = (E - e*sin(E) - M_e)/(1 - e*cos(E));
    E = E - delta;
end
```

Now calculate the position if necessary

```
r = a*(1-e*cos(E)); % [km]
theta = acos((((a*(1-e^2))/r)-1)/e);
% Get the position and velocity in the perifocal frame
r_p = r*cos(theta);
r_q = r*sin(theta);

v_p = (mu/h)*(-sin(theta));
v_q = (mu/h)*(e+cos(theta));
```

Published with MATLAB® R2019a