Table of Contents

```
%-------- HW 1 MATLAB code --------%
% Romeo Perlstein, section 0101 %

% Constants
mew_sun = 1.32712 * (10^11);
tall_er_ant = (10^-13);
step_size = 10000;
max_time = 70000000;

% Time step
t = [0:step_size:max_time];

% ODE options
ODE_options = odeset("RelTol", tall_er_ant, "AbsTol", tall_er_ant);

% format the data for 64 bit numeros
format long
```

Question 1

a)

From HW0

```
didymos_initial_x = -2.39573*10^8;
didymos_initial_y = -2.35661*10^8;
didymos_initial_z = 9.54384*10^6;
didymos_initial_vx = 1.24732*10^1;
didymos_initial_vy = -9.74427*10^0;
didymos_initial_vz = -8.78661*10^-1;
didymos_initial_state = [didymos_initial_x; didymos_initial_y;
 didymos_initial_z; didymos_initial_vx; didymos_initial_vy;
 didymos_initial_vz];

% Get didymos data
[T2,Y2] = ode45(@myodefun, t, didymos_initial_state, ODE_options, mew_sun);

% Get the specific energy
for i=1:1:(max_time/step_size+1) % I forget how to do it in one line my bad
    spef_energy(i) = (((sqrt(Y2(i,4)^2 + Y2(i,5)^2 + Y2(i,6)^2))^2)/2) -
 (mew_sun/sqrt(Y2(i,1)^2+Y2(i,2)^2+Y2(i,3)^2));
end
axes('FontSize', 16, 'NextPlot', 'add')
plot(T2, spef_energy, "-m")
title("Plot of the Specific Energy of Didymos over time", Units="normalize",
 Position=[.5,.9], FontSize=12)
xlabel("Time (s)")
ylabel("Specifc Energy (km^2/sec^2)")
axis equal
```

b)

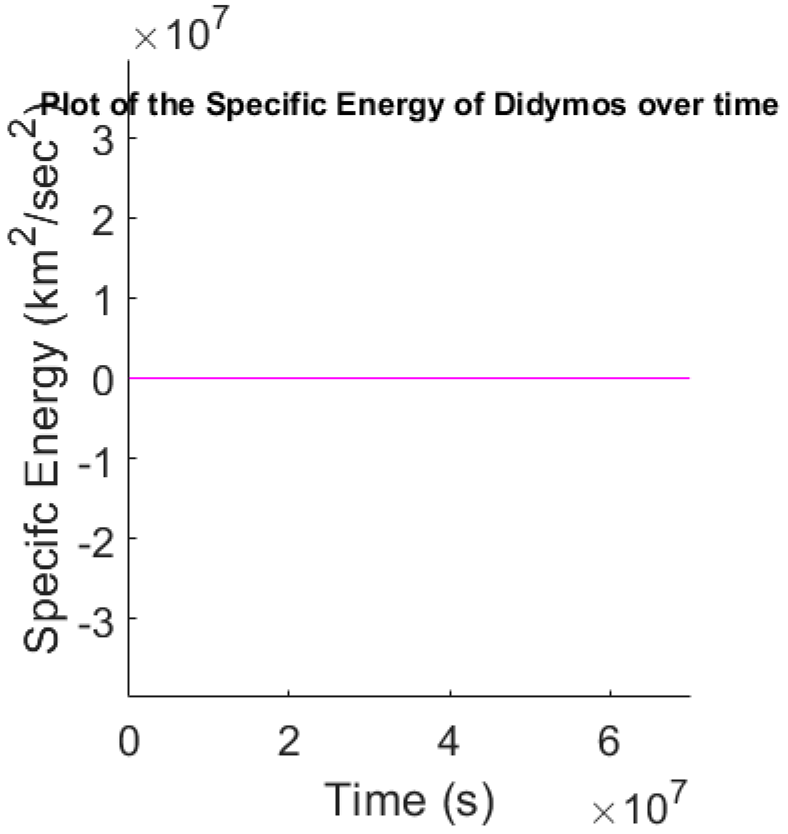
Get position and velocity vectors

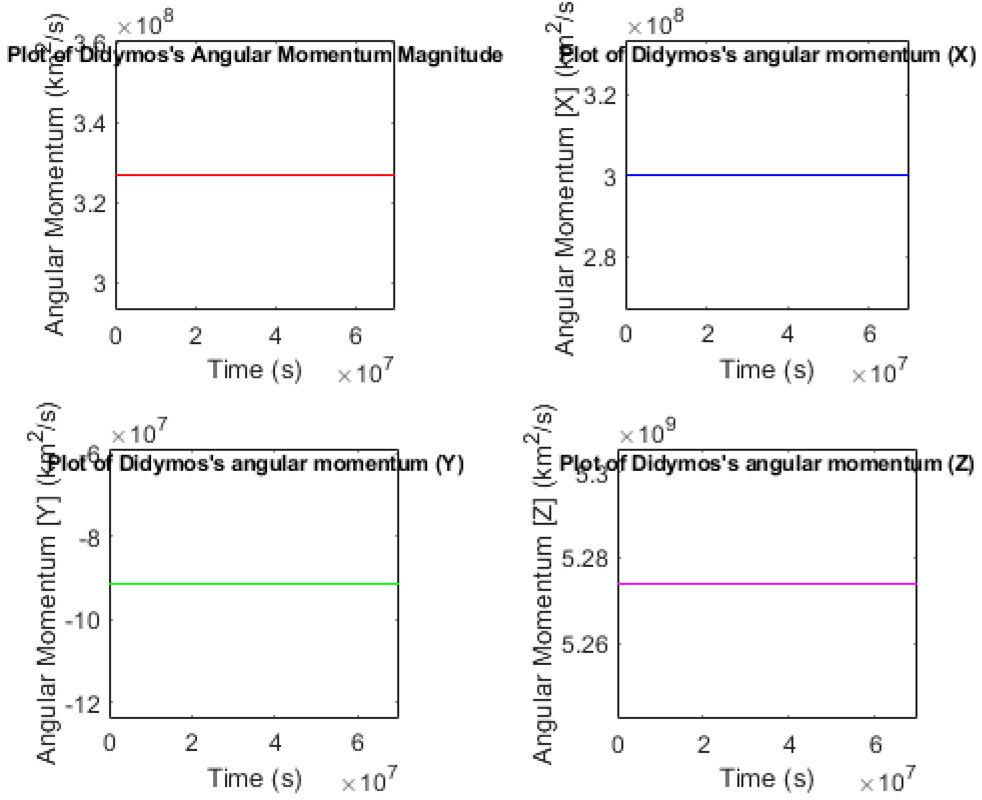
```
didymos_r = [Y2(:,1),Y2(:,2),Y2(:,3)];
didymos_v = [Y2(:,4),Y2(:,5),Y2(:,6)];

% Cross them bad boys (real)
didymos_h = cross(didymos_r,didymos_v);

% Get the magnitude
for i=1:1:(max_time/step_size+1) % I forget how to do it in one line my bad
    didymos_h_mag(i) = sqrt(didymos_h(i,1)^2 + didymos_h(i,2)^2 +
 didymos_h(i,2)^2);
end

figure

tiledlayout(2,2);
```

```
nexttile % tile 1
plot(T2,didymos_h_mag, "-r")
title("Plot of Didymos's Angular Momentum Magnitude", Units="normalize",
 Position=[.5,.9], FontSize=8)
xlabel("Time (s)")
ylabel("Angular Momentum (km^2/s)")
axis equal

nexttile % tile 2
plot(T2, didymos_h(:,1), "-b")
title("Plot of Didymos's angular momentum (X)", Units="normalize",
 Position=[.5,.9], FontSize=8)
xlabel("Time (s)")
ylabel("Angular Momentum [X] (km^2/s)")
axis equal

nexttile % tile 3
plot(T2, didymos_h(:,2), "-g")
title("Plot of Didymos's angular momentum (Y)", Units="normalize",
 Position=[.5,.9], FontSize=8)
xlabel("Time (s)")
ylabel("Angular Momentum [Y] (km^2/s)")
axis equal

nexttile % tile 4
plot(T2, didymos_h(:,3), "-m")
title("Plot of Didymos's angular momentum (Z)", Units="normalize",
 Position=[.5,.9], FontSize=8)
xlabel("Time (s)")
ylabel("Angular Momentum [Z] (km^2/s)")
axis equal
```

c)

From my understanding, the previous two plots indicate that my two-body-problem propagator is working properly because the momentum and energy are "conserved" (i.e., they are pretty much a straight line!). This makes sense due to the constraints we're applying. If energy or angular momentum were not conserved, the planet would eventually begin spiraling in or out or in some way, shape, or form and not form a perfect orbit.


Plot of the Specific Energy of Didymos over time
×10^7
Specifc Energy (km^2/sec^2)
3
2
1
0
-1
-2
-3
0
2
4
6
Time (s)
×10^7


Plot of Didymos's Angular Momentum Magnitude
×10^8
Angular Momentum (km^2/s)
3.6
3.4
3.2
3
0
2
4
6
Time (s)
×10^7
Plot of Didymos's angular momentum (X)
×10^8
Angular Momentum [X] (km^2/s)
3.2
3
2.8
0
2
4
6
Time (s)
×10^7
Plot of Didymos's angular momentum (Y)
×10^7
Angular Momentum [Y] (km^2/s)
-6
-8
-10
-12
0
2
4
6
Time (s)
×10^7
Plot of Didymos's angular momentum (Z)
×10^9
Angular Momentum [Z] (km^2/s)
5.3
5.28
5.26
0
2
4
6
Time (s)
×10^7

2

a)

all angles of true anomaly

b)

180 deg and 0 deg

c)

0 deg

d)

0 deg

3

```
alt = 6000;
v = 8.5; % km/s
flight_path_ang = 0.5; % degrees
mew = 3.986*10^5; % km3/s2
earth_radius = 6378; % Radius of Earth
r = earth_radius + alt; % total radius value

spef_energy3 = (v^2)/2 - mew/r; % specific energy
spef_ang_momentum3 = r*v*cosd(flight_path_ang); % specific angular momentum
eccen = sqrt(1+(2*spef_energy3*(spef_ang_momentum3)^2)/(mew^2)); %
 eccentricity

semi_major_axis = -mew/(2*spef_energy3); % semi-major axis

rp = semi_major_axis*(1-eccen);
```

a)

The eceentricity came out to be 1.2436, meaning that e > 1. With that case, Luke's orbit should be a hyperbolic conic section!

b)

```
semi_major_axis
% The semi-major axis of this orbit is at #50806.7696078 km (which checks
% out since it is a hyperbola, and the semi-major axis is located on the
% outside of it!)
```

c)

```
spef_ang_momentum3
% The specific angular momentum of Luke’s orbit is 105208.993811 km2/s
```

d)

```
eccen
% As stated in part a), the eccentricity of this orbit is 1.2436
```

e)

```
rp
% The radius of the periapsis is 12377.1497095 km
```

```
semi_major_axis =

    -5.080676960781794e+04


spef_ang_momentum3 =

     1.052089938113507e+05


eccen =

   1.243612215557545


rp =

     1.237714970948228e+04
```

4

```
a4 = 20000;
e4 = 0.4;
mew = 3.986*10^5; % km3/s2
```

a)

```
anom = 30; % in degrees

p4 = a4*(1-(e4^2)); % get P value
r4 = p4/(1+e4*cosd(anom)) % Find radius at specified anomaly
```

b)

```
anom2 = 330;
r4_2 = p4/(1+e4*cosd(anom2)) % Find radius at specified anomaly
```

c)

using r4 for anom = 30 degrees

```
v4 = sqrt(2*(-mew/(2*a4) + mew/r4)) % km/s
```

d)

using r4_2 for anom = 330 degrees

```
v4_2 = sqrt(2*(-mew/(2*a4) + mew/r4_2)) % km/s
```

e)

```
h4 = sqrt(mew*p4);
flight_ang = acosd(h4/(r4*v4)) % flight angle at anom = 30 degrees
```

f)

```
flight_ang_2 = acosd(h4/(r4_2*v4_2)) % flight angle at anom = 330 degrees
```

g)

```
apoapsis = a4*(1+e4)
```

h)

```
v4_3 = sqrt(2*(-mew/(2*a4) + mew/apoapsis))
```

```

r4 =

     1.247762418928247e+04


r4_2 =

     1.247762418928247e+04


v4 =

   6.630261525936095
```

```
v4_2 =

   6.630261525936095


flight_ang =

   8.449113362178327


flight_ang_2 =

   8.449113362178327


apoapsis =

       28000


v4_3 =

   2.922572252559134
```

5

```
rp5 = 10000;
e5 = 1;
mew = 3.986*10^5; % km3/s2
```

a)

```
v5 = sqrt((2*mew)/rp5)
```

b)

The apoapsis of a parabolic orbit is infinity!

c)

This orbit's conic is parabolic!

```
% figure
% % Plot Sun
% surf(X_2,Y_2,Z_2)
% hold on


v5 =
   8.928605714219886
```

ODE function handles

From ENAE301

```
function ydot = myodefun(t, y, mew)
    r_mag = norm(y(1:3));
    ydot(1,1) = y(4);
    ydot(2,1) = y(5);
    ydot(3,1) = y(6);
    ydot(4,1) = (-mew/r_mag^3)*y(1);
    ydot(5,1) = (-mew/r_mag^3)*y(2);
    ydot(6,1) = (-mew/r_mag^3)*y(3);
end
```

Published with MATLAB® R2022b