Table of Contents


```
% Hw 1 - A351 - Joshua Oates
```

0

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

1

```
%tString = "2000-01-1 12:00:00"; % heart check, should come out to known value
tString = "2022-09-22 4:00:00";
t = datetime(tString);
jd1 = juliandate(t); % MATLABs JD
jd2 = joshJulian(t); % My JD and J2000

disp(tString+" in julian date according to MATLAB is "+string(jd1)+" days.")
disp(tString+" in julian date according to my function is "+string(jd2)+"
 days.")

clear tString t
```

```
2022-09-22 4:00:00 in julian date according to MATLAB is 2459844.6667 days.
2022-09-22 4:00:00 in julian date according to my function is 2459844.6667
 days.
```

2

```
tString1 = "2010-12-21 10:00:00"; %UT
tString2 = "2022-07-04 12:30:00";

t1 = datetime(tString1);
t2 = datetime(tString2);

longitude1 = 144.966667; %144 58 e
longitude2 = 360-120.653;%120.653 w

[~,theta1] = joshJulian(t1,longitude1);
[~,theta2] = joshJulian(t2,longitude2);
```

```
disp(tString1+" in local sideral time is "+string(theta1)+" degrees.")
disp(tString2+" in local sideral time is "+string(theta2)+" degrees.")

clear tString2 tString1 t2 t1 longitude2 longitude1
```

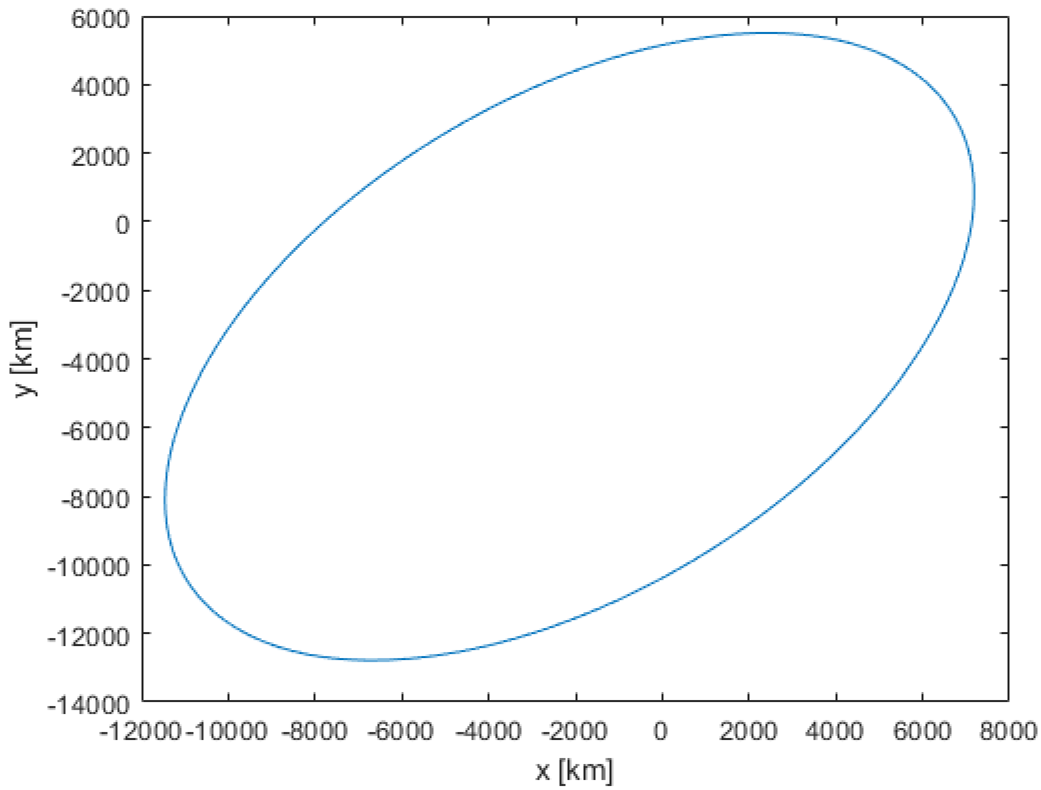
2010-12-21 10:00:00 in local sideral time is 24.8341 degrees.
2022-07-04 12:30:00 in local sideral time is 349.3495 degrees.

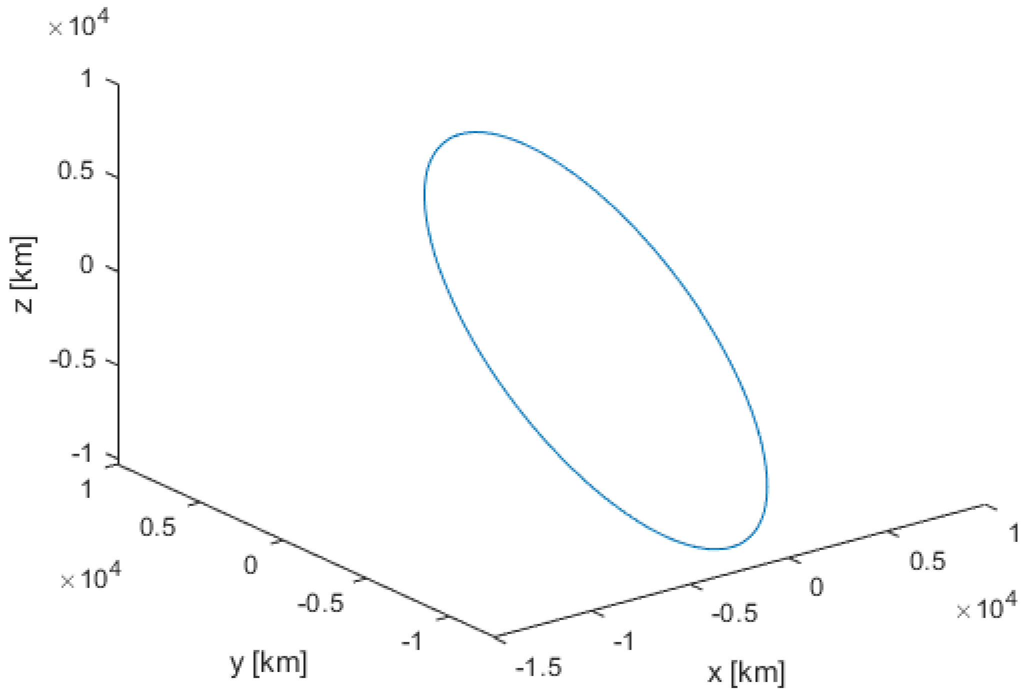
3

```
mu_e = 398600;
r0_v = [3207,5459,2714];
v0_v = [-6.532,.7835,6.142];
Y0 = [r0_v, v0_v];

tspan = [0, 24*3600];

options = odeset('RelTol', 1e-8,'AbsTol',1e-8);

[~, Yout] = ode45(@orbit, tspan, Y0, options, mu_e);

figure
plot(Yout(:,1),Yout(:,2))
xlabel('x [km]')
ylabel('y [km]')

figure
plot3(Yout(:,1),Yout(:,2),Yout(:,3))
xlabel('x [km]')
ylabel('y [km]')
zlabel('z [km]')


rf_v = Yout(end,1:3);
vf_v = Yout(end,4:6);
rf = norm(rf_v);
vf = norm(vf_v);

disp("At t+24:00:")

myString = "<";
for i=1:length(vf_v)
    myString = myString+vf_v(i);
    if i < length(vf_v-1)
        myString = myString+" , ";
    end
end
myString = myString+">";


disp("The velocity vector is "+myString+" km/s")
disp("The speed is "+string(vf)+" km/s")
```

```
myString = "<";
for i=1:length(rf_v)
    myString = myString+rf_v(i);
    if i < length(rf_v-1)
        myString = myString+" , ";
    end
end
myString = myString+">";

disp("The position vector is "+myString+" km")
disp("The distance is "+string(rf)+" km")
disp("Below is the orbit visualized both flat and 3d.")

clear options r0_v v0_v Yout tspan y0
```

At t+24:00:
The velocity vector is <-7.1527 , -3.3316 , 2.6179> km/s
The speed is 8.3135 km/s
The position vector is <-1991.4383 , 4408.9711 , 5961.1807> km
The distance is 7677.2734 km
Below is the orbit visualized both flat and 3d.





4

```
clear all

syms t ecc mu h
assume(t,"real")
assumeAlso(t >= 0)
assumeAlso(t <= 2*pi)
assume(mu, "real")
assumeAlso(mu>0)
assume(h,"real")
assumeAlso(h>0)
assume(ecc,"real")
assumeAlso(ecc > 0)
assumeAlso(ecc < 1)

v_az = (mu/h)*(1+ecc*cos(t));
v_ra = (mu/h)*ecc*sin(t);
RHS = sqrt(v_az^2+v_ra^2);
RHS = simplify(expand(RHS));
LHS = (mu/h)*sqrt(1+2*ecc*cos(t)+ecc^2);
LHS = simplify(expand(LHS));

disp("Using symbolic toolbox, I was able to show that RHS =
 sqrt(v_az^2+v_ra^2) = LHS = (mu/h)*sqrt(1+2*ecc*cos(t)+ecc^2). This was
```

```
validated using the 'isAlways' prover. Note: resonable assumpetions were made
about the values of the variables in the equation.")
disp("below is the result of isAlways(RHS==LHS)")
isAlways(RHS==LHS)

clear all
```

```
Using symbolic toolbox, I was able to show that RHS = sqrt(v_az^2+v_ra^2)
 = LHS = (mu/h)*sqrt(1+2*ecc*cos(t)+ecc^2). This was validated using the
 'isAlways' prover. Note: resonable assumpetions were made about the values of
 the variables in the equation.
below is the result of isAlways(RHS==LHS)

ans =

  logical

   1
```

5

```
mu_m = 42828; %km^3/s^2
r_m = 3390; %km
z = 200; %km
r = z+r_m; %km
RHS = sqrt(mu_m/r);

Period = 2*pi*sqrt((r^3)/mu_m)/3600;
disp("The period of the orbiting body is "+string(Period)+" days.")
```

```
The period of the orbiting body is 1.8141 days.
```

6

```
clear all;
mu_e = 398600;%km^3/s^2
r_e = 6378;%km
rp = 10000;%km
ra = 100000;%km

ecc = (ra-rp)/(ra+rp);
a= (ra+rp)/2;
P = ((2*pi)/sqrt(mu_e))*a^1.5;
P=P/3600;
h = (a*mu_e*(1-ecc^2))^.5;

vr  = @ (theta)  (mu_e/h)*ecc*sin(theta);
vaz = @ (theta)  (mu_e/h)*(1+ecc*cosd(theta));
v_p = vaz(0);
v_a = vaz(180);

r_z10k = r_e+10000;
```

```
theta=acosd((1/(r_z10k*(mu_e/h^2))-1)/ecc);
vaz_10k = vaz(theta);
vr_10k = vr(theta);

disp("For the given orbit the following values were found")
disp("ecc is "+string(ecc))
disp("a is   "+string(a)+" km")
disp("P is   "+string(P)+" hrs")
disp("h is   "+string(h)+" m^2/s^2")
disp("theta at 10000 km is "+string(theta)+" degrees")
disp("The azumth speed at 10000 km is  "+string(vaz_10k)+" km/s")
disp("The radial speed at 10000 km is  "+string(vr_10k)+" km/s")
disp("The speed at perigee is          "+string(v_p)+" km/s")
disp("The speed at apogee is           "+string(v_a)+" km/s")
```

```
For the given orbit the following values were found
ecc is 0.81818
a is   55000 km
P is   35.6577 hrs
h is   85130.9152 m^2/s^2
theta at 10000 km is 82.2638 degrees
The azumth speed at 10000 km is  5.1979 km/s
The radial speed at 10000 km is  2.1072 km/s
The speed at perigee is          8.5131 km/s
The speed at apogee is           0.85131 km/s
```

Published with MATLAB® R2022a

```
function [jd,thetaTime ,j2000, j0, ut, thetaG] = joshJulian(t,thetaLongitude)
% takes t as a datetime object in UT and a longitude
% jd - juliandate day
% thetaTime - local sidereal time in degrees
% j2000 - julian date from 2000
% j0 - julian days
% ut - UT in hours
% thetaG - Grennich sidereal time

arguments
    t (1,1) datetime
    thetaLongitude (1,1) {mustBeReal} = 0
end


j2000_0 = 2451545;

[yr,mo,da] = ymd(t);
[hr,mn,sc] = hms(t);

j0 = 367*yr-floor((7*(yr+floor((mo+9)/12)))/4)+floor((275*mo)/9)+da+1721013.5;
ut = hr + mn/60 + sc/3600;
jd = j0 + (ut/24);
j2000 = jd - j2000_0;

t0 = (j0 - j2000_0)/36525;

thetaG0 = 100.4606184 + 36000.77004*t0 + 0.000387933*t0^2 -
 2.583*(10e-8)*t0^3;
thetaG = thetaG0 + 360.98564724* (ut/24);
thetaTime = thetaG + thetaLongitude;

thetaG = mod(thetaG,360);
thetaTime = mod(thetaTime,360);
end
```

Published with MATLAB® R2022a

```
function someDerivatives = orbit(t,Y,mu_e) % y is current state, t is time
 (unused), mu is passed in
x(1) = Y(1);
x(2) = Y(2);
x(3) = Y(3);

dx(1) = Y(4);
dx(2) = Y(5);
dx(3) = Y(6);

r = norm(x);

ddx(1) = -mu_e*x(1)/r^3;
ddx(2) = -mu_e*x(2)/r^3;
ddx(3) = -mu_e*x(3)/r^3;

someDerivatives = [dx' ; ddx'];
end
```

Published with MATLAB® R2022a