
Table of Contents

Josh O	1
4.15	1
5.6	2
6.8	3
6.23	4
6.25	5
6.31	5
6.44	5
6.47	6
dependancies	7
functions	7

Josh O

```
clear all
close all
clc

addpath('C:\joshFunctionsMatlab\')

% 4.15
% % % 5.6 - lambert
% 6.8
% 6.23
% 6.25
% 6.31 - rotation
% % 6.44 - HT and inc
% 6.47 - ODE

disp("----- HW2 - Josh Oates -----")
```

4.15

```
[Cx,Cy,Cz]=joshAxisRotation();

mu_e = 398600;
r_e = 6378; % km

ecc = 1.5;
zpapse = 300; % km

inc = 35; % degrees
raan = 130; % degrees
aop = 115; % degrees
```

```

inc = deg2rad(inc);
raan = deg2rad(raan);
aop = deg2rad(aop);
rpapse = zpapse + r_e;
theta = 0;
h = sqrt(mu_e*rpapse*(1+ecc*cos(theta)));

vpapse = h/rpapse;

rp = [1,0,0]*rpapse;
vp = [0,1,0]*vpapse;
clear vpapse rpapse zpapse

Q = Cz(raan)*Cx(inc)*Cz(aop);
Q = Q'; % Perifocal -> ECI

reci = rp*Q;
veci = vp*Q;

disp("-----P4.15-----")
disp("My calculations have the following results:")
disp("Velocity in Perifocal Frame [km/s]:")
disp(vp)
disp("Position in Perifocal Frame [km]:")
disp(rp)
disp("Velocity in ECI Frame [km/s]:")
disp(veci)
disp("Velocity in Perifocal Frame [km]:")
disp(reci)
disp("H/C: norms of r and v in either frame should be equal:")
disp("Norm veci: "+string(norm(veci))+ " Norm vp: "+string(norm(vp)));
disp("Norm reci: "+string(norm(reci))+ " Norm rp: "+string(norm(rp)));

```

5.6

```

clear all

coefs = 15;
[Cc,Sc]=joshStumpffCoeffs(coefs);
C = @(z) sum(Cc.*joshStumpffZ(z,coefs));
S = @(z) sum(Sc.*joshStumpffZ(z,coefs));

mu_e = 398600;

r1 = [5644,2830,4170]; % km
r2 = [-2240,7320,4980]; % km
dt = 20; % min
dt = dt*60; % sec

z = cross(r1,r2);
z = z(3);

```

```

theta1 = acos(dot(r1,r2)/(norm(r1)*norm(r2)));
theta2 = 2*pi - acos(dot(r1,r2)/(norm(r1)*norm(r2)));

[fz,y,A,z,flag,glag,gdotlag] =
    joshfLambert(norm(r1),norm(r2),dt,theta1,mu_e,Cc,Sc);

v1 = (1/glag)*(r2-flag*r1);
v2 = (1/glag)*(gdotlag*r2-r1);

disp("-----P5.6-----")
disp("My calculations have the following results:")
disp("V1 in km/s: ")
disp(v1)
disp("V2 in km/s: ")
disp(v2)
disp("H/C: We should be using the short side and theta1 < theta2 so this makes
    sense")

```

6.8

```

clear all;

mu_e = 398600;
r_e = 6378; % km

z1 = 300; % km
z3 = 3000; % km
r1 = r_e+z1;
r3 = r_e+z3;

v1 = sqrt(mu_e/r1);
v3 = sqrt(mu_e/r3);

ecc2 = (r3-r1)/(r3+r1);
a2 = (r3+r1)/2;

h2 = sqrt(a2*mu_e*(1-ecc2^2));

vp2 = h2/r1;
va2 = h2/r3;

dv = abs(va2-v3)+abs(vp2-v1);
P2 = (2*pi/sqrt(mu_e))*a2^1.5; % sec
tranfertime = P2/2;
tranfertime = tranfertime/60;

disp("-----P6.8-----")
disp("My calculations have the following results:")
disp("dv [km/s]: "+string(dv))
disp("transit time [s]: "+string(tranfertime))
disp("H/C: transfer time is half of the transfer period. This period makes
    sense for something LEO MEOish.")

```

6.23

```
clear all

mu_e = 398600;
r_e = 6378; % km

theta1 = 150; % degrees
theta1 = deg2rad(theta1);
theta2 = 45; % degrees
theta2 = deg2rad(theta2);
ra1 = 18900; % km
rp1 = 8100; % km
a1 = (ra1+rp1)/2;
P1 = (2*pi/sqrt(mu_e))*a1^1.5; % sec

ecc1 = (ra1-rp1)/(ra1+rp1);
Me1 = joshAnomalyCalculator(ecc1,theta1);
n1 = sqrt(mu_e/a1^3);
t1 = Me1/n1; % time since periapse orbit 1 object B

Me2 = joshAnomalyCalculator(ecc1,theta2);
t2 = Me2/n1; % time since periapse orbit 1 object C

P2 = (P1-t1); % time till periapse object c
P2 = P2+t2; % time till rendezvous

h1 = sqrt(a1*mu_e*(1-ecc1^2));
r1_45 = (h1^2/mu_e)/(1+ecc1*cos(theta2)); % at theta 45 what is the radius of
both orbits
rp2 = r1_45; % at a theta of 45, satalite B makes its location its new
periapse

a2 = (P2*(sqrt(mu_e)/(2*pi)))^(2/3);
ra2 = a2*2-rp2;
ecc2 = (ra2-rp2)/(ra2+rp2);
h2 = sqrt(a2*mu_e*(1-ecc2^2));
vp2 = h2/rp2;
%
[vaz1_45,vr1_45] = joshVazVr(theta2,ecc1,h1,mu_e); % original vaz and vr

vr2 = 0;
vaz2 = vp2;

% v1_45 = sqrt(vaz1_45^2+vr1_45^2)
vp2 = [vaz2,vr2];
v1_45 = [vaz1_45,vr1_45];
dv = norm(v1_45-vp2)*2;

disp("-----P6.23-----")
disp("My calculations have the following results:")
disp("dv [km/s]: "+string(dv));
```

```
disp("H/C: The orbital period used in the calculation makes sense for a MEO
orbit.")
```

6.25

```
clear all
mu_e = 398600;
r_e = 6378; % km
theta1 = deg2rad(100);

rp1 = 1270+r_e;
vp1 = 9;
[a1,ecc1,~,~,~,~,h1,T1,E1] = joshCOE(rp1,vp1,mu_e,"magnitude");

[vaz1,vr1,gamma1] = joshVazVr(theta1,ecc1,h1,mu_e);

r100 = h1^2/(mu_e*(1+ecc1*cos(theta1)));
ecc2 = .4;
a2 = r100*(1+ecc2*cos(theta1))/(1-ecc2^2);

h2 = sqrt(a2*mu_e*(1-ecc2^2));

[vaz2,vr2,gamma2] = joshVazVr(theta1,ecc2,h2,mu_e);

dgamma = rad2deg(gamma2-gamma1);
dv = norm([vaz2, vr2]-[vaz1, vr1]);

disp("-----P6.25-----")
disp("My calculations have the following results:")
disp("Delta gamma [degrees]: "+string(dgamma))
disp("Delta v [km/s]: "+string(dv))
disp("H/C: We would imagine a moderate delta v for a maneuver like this. This
seems to make sense for a small apseline rotation.")
```

6.31

on paper

6.44

```
clear all
mu_e = 398600;
r_e = 6378; % km

r1 = 300+r_e;
r2 = 600+r_e;

v1 = sqrt(mu_e/r1);
v2 = sqrt(mu_e/r2);

inc = deg2rad(20);
```

```

[dvh2,dvh1,dvh,~,~,~,vt1,vt2] = joshHomann(r1,v1,r2,v2,mu_e);

dvi = sqrt(2*(v2^2)-2*(v2^2)*cos(inc));
dva = dvi+dvh;

dvh1i = joshLawCos(v1,vt1,inc);
dvh2i = joshLawCos(v2,vt2,inc);

dvb = dvh1 + dvh2i;
dvc = dvh2 + dvh1i;

disp("-----P6.44-----")
disp("My calculations have the following results:")

disp("A) dv [km/s]: "+string(dva))
disp("B) dv [km/s]: "+string(dvb))
disp("C) dv [km/s]: "+string(dvc))

disp("H/C: We would expect the lowest delta v to be a combined maneuver at a
      greater altitude, this way there is less velocity to change so to speak and a
      small dv will create a larger inc change.")

```

6.47

```

clear all
mu_e = 398600;
r_e = 6378; % km

r0 = [436;6083;2529];
v0 = [-7.340;-.5125;2.497];
m0 = 1000;
X0 = [r0;v0;m0];

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

tspan = [0,89]*60;%s
[t1,X1] = ode45(@XdotFunCoast,tspan,X0,options);

tspan = [0,120];
X0 = X1(end,:);
[t2,X2] = ode45(@XdotFunBurn,tspan,X0,options);

X = [X1;X2];
t = [t1;t2+t1(end)];

v3 = X(end,4:6);
r3 = X(end,1:3);

% [a,ecc,theta,inc,raan,aop,h,T,E] = joshCOE(r3,v3)

X0 = X2(end,:);
tspan = [0,200]*60;%s

```

```

[t3,X3] = ode45(@XdotFunCoast,tspan,X0,options);

X = [X1;X2;X3];
t = [t1;t2+t1(end);t3+t2(end)+t1(end)];

close all
figure
hold on
plot3(X1(:,1),X1(:,2),X1(:,3))
plot3(X2(:,1),X2(:,2),X2(:,3))
plot3(X3(:,1),X3(:,2),X3(:,3))
title("Simple graph H/C to make sure ODEs worked")

n = length(X);
rmag = zeros(n,1);
for i = 1:n
    rmag(i) = norm([X(i,1),X(i,2),X(i,3)]);
end

[rmax,i] = max(rmag);
zmax = rmax - r_e;
tmax = t(i);
tmax = tmax/60;

disp("-----P6.44-----")
disp("My calculations have the following results:")
disp("Max altitude [km]: "+string(zmax))
disp("Max altitude time since t0 [min]: "+string(tmax))
disp("H/C: the time to get to apogee seems to be within range for 2ish orbits
    in LEO which is what the included graph seems to predict.")

```

dependancies

```

disp("-----Dependancies-----")
disp("My code uses the following functions: ")
depends = matlab.codetools.requiredFilesAndProducts('C:\AERO351\A351HW3\HW3');
disp(depends')

```

functions

```

% 6.47 dX
function dX = XdotFunBurn (t,X)
    mu_e = 398600;
    isp = 300; % s
    g0 = 9.80665; % m/s^2
    g0 = g0/1000; % km/s^2
    T = 10000; % kg.m/s^2
    T = T/1000; % kg.km/s^2

    r = X(1:3);
    v = X(4:6);
    m = X(7);
    a = (-mu_e*r)/norm(r)^3 + (T/m)*(v/norm(v));

```

```

    dm = (-T/(isp*g0));

    dX = [v;a;dm];
end

function dX = XdotFunCoast (t,X)
    mu_e = 398600;
    isp = 300; % s
    g0 = 9.80665; % m/s^2
    g0 = g0/1000; % km/s^2
    T = 0; % kg.m/s^2
    T = T/1000; % kg.km/s^2

    r = X(1:3);
    v = X(4:6);
    m = X(7);
    a = (-mu_e*r)/norm(r)^3 + (T/m)*(v/norm(v));
    dm = (-T/(isp*g0));

    dX = [v;a;dm];
end

% for lamberts
function out = fp_zcheck(z,fp,fpz0)
    if z == 0
        out = fpz0;
    else
        out = fp;
    end
end
end

```

Published with MATLAB® R2022a