## Table of Contents

```
% Joshua Oates - AERO351 - HW2


% Feel free to use MATLAB (or similar product) when appropriate but please
 submit a pdf/html
% of the published code.
%
% Chapter 2 problems: 2.23, 2.36, 2.37
%
% Chapter 3 problems: 3.8, 3.10, and 3.20 (check this answer with using ODE45
 as well, do they
% agree?)
%
% Chapter 4 problems: 4.5, 4.7
```

## 0

```
clear all;
close all;
clc

addpath('C:\joshFunctionsMatlab\')

disp("------------ HW1 - Josh Oates ------------")
```

## 2.23 find gamma(theta) and z(theta)

```
mu_e = 398600;
r_e = 6378; % km
zp = 500; % km
rp=zp+r_e; % km
v = 10; % km/s
theta = 120; % degrees
theta = deg2rad(theta); % rad
```

```
[a,ecc,~,~,~,~,h,T]=joshCOE(rp,v,mu_e); % COEs
[vaz,vr,gamma] = joshVazVr(theta,ecc,h,mu_e); % Velocity components
gamma = rad2deg(gamma);
T = T/3600;

r=a*((1-ecc^2)/(1+ecc*cos(theta)));
z120=r-r_e;
disp("------------P2.23------------")
disp("My calculations have the following results:")
disp("Flight path angle (gamma): "+string(gamma)+" degrees")
disp("H/C: Theta is between 0 and 180, so gamma should be positive")
disp("Altitude at theta = 120 degrees: "+string(z120)+" km")
disp("H/C: Altitude is in LEO range")
```

## 2.36 excess V

```
clear all;
mu_e = 398600;
r_e = 6378; % km
zp = 250; % km
rp=zp+r_e; % km
v = 11; % km/s
theta = 100; %degrees
theta = deg2rad(theta);% rad


[a,ecc,~,~,~,~,h,T,E]=joshCOE(rp,v,mu_e);
v_inf = sqrt(mu_e/-a);
r100 = (h^2/mu_e)/(1+ecc*cos(theta));
z100 = r100-r_e;
[vaz,vr,gamma] = joshVazVr(theta,ecc,h,mu_e);

disp("------------P2.36------------")
disp("My calculations have the following results:")
disp("Excess escape velocity: "+string(v_inf)+" km/s")
disp("H/C: Hyperbolic tragector should have positive excess escape velocity")
disp("r at theta = 100 degrees: "+string(r100)+" km")
disp("H/C: z > 250")
disp("Azmuthal velocity at theta = 100 degrees: "+string(vaz)+" km/s")
disp("H/C: Azmuthal velocity should be positive")
disp("Radial velocity at theta = 100 degrees: "+string(vr)+" km/s")
disp("H/C: At theta = 100 object is on departure so radial velocity should be
 positive")
```

## 2.37 meteroid

```
clear all;
mu_e = 398600;
r_e = 6378; % km
r0=402000;%km
theta=150;%deg
theta=deg2rad(theta);
v0 = 2.23;%km/s
```

```
E=(v0^2/2)-(mu_e/r0); % Energy
a = mu_e/(2*E);

syms h % set up h for symbolic solving
assume(h,'real')
assumeAlso(h>0)
eqn = sym(r0== (h^2/mu_e)*(1+(2*E*(h^2/mu_e^2)+1)^.5*cos(theta))^-1); %
 symbolic equation to solve for h
h = solve(eqn,h);
h = double(h);
ecc = (1/(r0*(mu_e/h^2))-1)/cos(theta); % solve for ecc

rp = a*(ecc-1);
zp = rp-r_e;

v = h/rp;

disp("------------P2.37------------")
disp("My calculations have the following results:")
disp("Eccentricity: "+string(ecc))
disp("H/C: For hyperbolic ecc should be above 1")
disp("Altitude at periapse: "+string(zp)+" km")
disp("H/C: Altitude at closest approach is lower than initial velocity but
 high enough to orbit")
disp("Velocity at closest approach: "+string(v)+" km/s")
disp("H/C: v at closest approach is higher than v initial")

% [a,ecc,~,~,~,~,h,T,E]=joshCOE(ri,vi,mu_e);
```

## 3.8 time above 400

```
clear all;
mu_e = 398600;
r_e = 6378; % km
ra = r_e + 600;
rp = r_e + 200;
a = (rp+ra)/2;
ecc = (ra-rp)/(rp+ra); % definition of ecc

h = sqrt(a*(1-ecc^2)*mu_e); % solve for h

r = 400 +r_e; % solve r
theta = acos((1/(r*(mu_e/h^2))-1)/ecc); % solve theta

[Me,Ean]=joshAnomalyCalculator(ecc,theta); % convert TA to Me
P=((2*pi)/sqrt(mu_e))*a^1.5;
n=sqrt(mu_e/a^3); % definition n
tsp = Me/n;
t400k = P-2*tsp; % time above 400 km
t400k = t400k/60;

disp("------------P3.8------------")
disp("My calculations have the following results:")
disp("Time spent over 400 km: "+string(t400k)+" min")
disp("H/C: time over 400 km is less than period")
disp("Intermediate H/C used:")
disp("      ecc < 1")
disp("      period resonable for LEO")
```

## 3.10

```
clear all;
mu_e = 398600;
rp = 10000; % km
P = 14*60*60;
a = ((sqrt(mu_e)/(2*pi))*P)^(2/3); % equation for a
ra = a*2-rp;
ecc = (ra-rp)/(rp+ra);
h = sqrt(a*(1-ecc^2)*mu_e);
v0= h/rp; % this is because v0 is at periapse and is perpendicular to r at
 periapse
r0 = rp;
pr = 0; % dot product of v0 and r0 as vectors, 0 b/c they are perpendicular

dt = 10*60*60;
coefs = 15; % 15 terms in stumpff functions
[Cc,Sc]=joshStumpffCoeffs(coefs); % generate stumpff coeffecients
C = @(z) sum(Cc.*joshStumpffZ(z,coefs)); % generate stumpff functions
S = @(z) sum(Sc.*joshStumpffZ(z,coefs));
[fX,fpX]=joshfChi(r0,v0,mu_e,a,dt,Cc,Sc,pr); % generate f and fp which can be
 put into a newton solver for X
X0 = sqrt(mu_e)*dt*abs(1/a); % initial guess
[X] = joshNewtons(fX,fpX,X0,1e-14); % newton solver for X

Ean = X/sqrt(a); % Eccentric anomaly
theta = 2*atan(tan(Ean/2)/sqrt((1-ecc)/(1+ecc)));
% [Me,Ean2] = joshAnomalyCalculator(ecc,theta)
% [~,Ean2]=joshQuadrant(Ean2)

r = (h^2/mu_e)/(1+ecc*cos(theta)); % radial distance
[vaz,vr] =joshVazVr(theta,ecc,h,mu_e); % azmuthal and radial velocity
speed = sqrt(vaz^2 + vr^2);
disp("------------P3.10------------")
disp("My calculations have the following results:")
disp("radial position: "+string(r)+" km")
disp("H/C: r is between rp and ra")
disp("speed: "+string(speed)+" km/s")
disp("H/C: MEO orbit resonable velocity seems beleivable")
disp("radial velocity: "+ string(vr)+" km/s")
disp("H/C: since period is 14hrs and this is 10hrs in, the orbit is past
 apoapse, so radial velocity should be negative")
```

## 3.20

```
clear all;
```

```
mu_e = 398600;
dt = 2*60*60;
r0 = [20 -105 -19]*1000; % km
v0 = [.9 -3.4 -1.5]; % km/s
[a]=joshCOE(r0,v0,mu_e);


%%%%%%%%% same as above %%%%%%%%%
coefs = 15;
[Cc,Sc]=joshStumpffCoeffs(coefs);
C = @(z) sum(Cc.*joshStumpffZ(z,coefs));
S = @(z) sum(Sc.*joshStumpffZ(z,coefs));
[fX,fpX]=joshfChi(r0,v0,mu_e,a,dt,Cc,Sc);
X0 = sqrt(mu_e)*dt*abs(1/a);
[X] = joshNewtons(fX,fpX,X0,1e-14);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


f = 1-(X^2/norm(r0))*C(X^2/a); % f and g functions are possible because v0 and
 r0 are vectors
g = dt-((1/sqrt(mu_e))*X^3*S(X^2/a));
r = f*r0+g*v0; % position
f_dot = (sqrt(mu_e)/(norm(r)*norm(r0)))*X*((X^2/a)*S(X^2/a)-1);
g_dot = 1-(X^2/norm(r))*C(X^2/a);
v = f_dot*r0 + g_dot*v0; % velocity

shouldBe1 = f*g_dot-f_dot*g; % H/C
disp("------------P3.20------------")
disp("My calculations have the following results:")
disp("The final position vector in km:")
disp(r)
disp("The final velocity vector in km/s:")
disp(v)
disp("H/C: f*g_dot-f_dot*g value: "+string(shouldBe1))
```

## 4.5

```
clear all;
mu_e = 398600;
R = [6.5 -7.5 -2.5]*1000;
V = [4 3 -3];
[a,ecc,theta,inc,raan,aop,h,T,E] = joshCOE(R,V,mu_e); % COEs
inc = rad2deg(inc);
raan = rad2deg(raan);
aop = rad2deg(aop);
theta = rad2deg(theta);
disp("------------P4.5------------")
disp("My calculations have the following results:")
disp("Semimajor axis: "+string(a)+" km")
disp("Eccentricity: "+string(ecc))
disp("True anomaly: "+string(theta)+" degrees")
disp("Inclination: "+string(inc)+" degrees")
disp("Right ascention of ascending node: "+ string(raan)+" degrees")
disp("Argument of periapse: "+string(aop)+" degrees")
disp("h: "+string(h)+"m^2/s")
```

## 4.7

```
clear all;
r = [-6.6 -1.3 -5.2]*1000;
ecc = [-.4 -.5 -.6];
mu_e = 398600;

theta = acos(dot(r,ecc)/(norm(ecc)*norm(r))); % solve for true anomaly
h = sqrt(mu_e*norm(r)*(1+norm(ecc)*cos(theta))); % use TA and get h
h = h*(cross(r,ecc)/norm(cross(r,ecc)));
inc = acos((dot([0 0 1],h))/norm(h)); % inclination degrees
inc = rad2deg(inc);

disp("------------P4.7------------")
disp("My calculations have the following results:")
disp("Inclination of this orbit: "+string(inc)+" degrees")
disp("H/C: r vector has a relatively large z component so this makes sense
 nominally")
```

## dependencies

```
depends = matlab.codetools.requiredFilesAndProducts("C:
\AERO351\A351HW2\HW2.m");
depends = depends';
%     {'C:\AERO351\A351HW2\HW2.m'                       }
%     {'C:\joshFunctionsMatlab\joshAnomalyCalculator.m'}
%     {'C:\joshFunctionsMatlab\joshCOE.m'               }
%     {'C:\joshFunctionsMatlab\joshIsOnes.m'            }
%     {'C:\joshFunctionsMatlab\joshNewtons.m'           }
%     {'C:\joshFunctionsMatlab\joshStumpffCoeffs.m'     }
%     {'C:\joshFunctionsMatlab\joshStumpffZ.m'          }
%     {'C:\joshFunctionsMatlab\joshVazVr.m'             }
%     {'C:\joshFunctionsMatlab\joshfChi.m'              }
```

*Published with MATLAB® R2022a*