% this is the main data function for ASTE 580
Project
% all rows ---> contain data of same planet in
order
% 1a,2e,3i,4raan,5argP,6M_0,7 p latus rectum
% standard convention is radians
% all coloums --indiviual paramerts earth venus
mars pluto
%
clc
clear all
more off
format shorte
disp('-----All distances are in Kms ,velocity in
km/s-------- ');

disp(' poe_d displays planetory parameters where
rows__Planets earth,venus,mars ,pluto and coloums
are a,e,i,RAAN,argP,M,p in degrees')
poe_d=[ 1.49597927e8 1.0820880e8 2.2794104e8
5.8902138e9;
   .016712113 .006776434 .09339833 .02488033;
   9.18702e-4 3.39474736 1.85029494 17.1671275;
   0 76.7004197 49.5802249 110.362856;
   102.91768 54.8701068 286.44747 114.258452;
   309.544036 196.058044 94.0519626 4.14798379]'
um_earth=3.986e5;
um_sun=1.327e11;
day_sec=3600*24;
rad_deg=180/pi;
% Converting the degrees in col 3 4 5 6 into
radians ...----doesnt contain
% p col7
deg_rad=pi/180;
poe_r=poe_d*deg_rad;
poe_r(1:4,1:2)=poe_d(1:4,1:2)

% Getting the JD for the events i.e JD1 epoch,JD2
launch ,JD3 venus
% flyby,JD4 mars flyby,JD 5 Pluto
disp('JD displays dates days in Julian dates in
order epoch,launch,venus flyby,mars flyby, pluto
arrival')

JD=[ 2448939.5 2453083.5 2453175.5 2453878.5
% Calculate ToF_epoch from epoch  total time elapsed since epoch 1..launch,2..venus, 
disp('tof_ep displays time elapsed in days since epoch for launch,venus flyby,mars flyby, pluto arrival')

tof_ep=JD(2:5)-JD(1)

% calculate tof between each date 1 venus arrival,2.mars flyby,3.. 
disp('tof_s displays time of flight Tof between each stage since launch,venus flyby,mars flyby, pluto arrival')

tof =tof_ep(2:4)-tof_ep(1:3)
tof_s=tof*3600*24

% Calculate equivalent M of each planet on all dates row--date each planet % coloumn different palnets
% Calulate n of each planet row 1 earth 2 venus 3 mars 4 pluto %coloumn at 1 launch,2 venus 3 mars 4 pluto
disp('displays n of each planet in order earth , venus, mars , pluto')

n=sqrt(um_sun./poe_r(1:4,1).^3)

for i=1:4
    for j=1:4
        n_dt(i,j)=n(i).*tof_ep(j);
    end
    n_dt;
end

n_dt
disp('n_dts shows time elapsed since epoch n(t-t_epoch) for respective dates');

n_dts=day_sec*n_dt

%Now Add M_epoch of respective planet to each value i.e rowl add M_0 %earth..
for i=1:4
for j=1:4
    M_tot(i,j)=poe_r(i,6)+n_dts(i,j);
end
end

disp('M_tot shows postion (M) of each planet(in row) on each date')
M_tot

%try extracting value of last revolutions actual from respective perigee
%row 1 earth coloum 1 launch,venus flyby,mars flyby,pluto (in radians)
format shorte
disp('M_end shows postion (M) of each planet(in row) on each date in last orbit/revolution')
M_rnd=M_extract(M_tot)

% Calculate the Eccentricity from given M_tof_ep and eccentricity of respective planet
for i=1:4
    for j=1:4
        E_tot(i,j)=newtonm(poe_r(i,2),M_tot(i,j));
    end
end
disp('M_tot shows postion (E) of each planet(in row) on each date')
E_tot

for i=1:4
    for j=1:4
        E_rnd(i,j)=newtonm(poe_r(i,2),M_rnd(i,j));
    end
end
disp('E_rnd shows postion (Eccentric anomaly) of each planet(in row) on each date in last orbit/revolution')
Ed_rnd=E_rnd*rad_deg
E_rnd2=M_extract(E_tot);
%Calculate the true anomaly to check
for i=1:4
    for j=1:4

\begin{verbatim}
f_tot(i,j)=newtonm_f(poe_r(i,2),M_tot(i,j));
end
end
disp('f_end shows position (true anomaly) of each planet (in row) on each date')
f_tot
for i=1:4
    for j=1:4
        f_rnd(i,j)=newtonm_f(poe_r(i,2),M_rnd(i,j));
    end
end
disp('f_end shows position (true anomaly) of each planet (in row) on each date in last orbit')
fd_rnd=f_rnd*rad_deg
f_rnd2=M_extract(f_tot);
% calulate the p latus rectum from a and e in given poe_d
disp('p is latus rectum of each planets orbit')
p=poe_d(1:4,1).*(1-(poe_d(1:4,2).^2))
disp('p added as seventh column to base data table poe_d')
poe_d(1:4,7)=p;
% to convert the orbita parameters at each stage into cartesian parameters
% r (1:1:k) contains position of earth at launch k1 X, K2 Y, K3Z
for i=1:4
    for j=1:4
        [ r(i,j,1:3) v(i,j,1:3)]=coe2rv(poe_d(i,7),poe_r(i,2),poe_r(i,3),poe_r(i,4),poe_r(i,5),f_rnd(i,j));
    end
end
disp('Below is orbital parameters converted into XYZ for each planet on each date')
r;
size(r);
\end{verbatim}
disp('Below is velocity parameters converted into XYZ for each planet on each date')

v;
% to calculate the magnitude of distances from sun at times launch, venus glyby...
% r_t and angles
r_sq=r.^2;
r_mag=r_sq(:,:,1)+r_sq(:,:,2)+r_sq(:,:,3);

disp('Distance from sun (length of r) of each planet on each date')

r_t=sqrt(r_mag); % this is the magnitude from the centre of the sun to planet at various times

format shorte
% to find angle between earth launch and venus flyby at centre(sun)
% r_all=squeeze(r(1:end,1:end,:));
disp('position and velocity of earth in HCI')

r1=squeeze(r(1,1,:)) % position of earth at launch 1 1
v1=squeeze(v(1,1,:))

disp('position and velocity of venus in HCI')

r2=squeeze(r(2,2,:))
v2=squeeze(v(2,2,:)) % position of venus at flyby

disp('position and velocity of mars in HCI')

r3=squeeze(r(3,3,:)) % position of mars at mars arrival
v3=squeeze(v(3,3,:))

disp('position and velocity of pluto in HCI')

r4=squeeze(r(4,4,:))
v4=squeeze(v(4,4,:))

disp('angle between earth and venus in HCI')

cos_theta12 = dot(r2,r1)/(norm(r1)*norm(r2))
% directly use this value to calculate C between two points
theta12=acos(cos_theta12)
theta12_d=theta12*rad_deg
disp('cos between venus and mars in HCI in radians')

\[
cos\theta_{23} = \frac{\text{dot}(r2, r3)}{\text{norm}(r3) \times \text{norm}(r2)}
\]
%directly use this value to calculate C between two points

disp('angle between venus and mars in HCI in radians')
disp('Manually correcting it to type 2 transfer due to anti clock wise rotation')
theta23 = 2*pi - acos(cos_theta23)

disp('in degrees')
theta23_d = theta23 * rad_deg

disp('cos between mars and pluto in HCI in radians')

\[
cos\theta_{34} = \frac{\text{dot}(r4, r3)}{\text{norm}(r4) \times \text{norm}(r3)}
\]
%directly use this value to calculate C between two points

disp('angle between mars and pluto in HCI in radians')
theta34 = acos(cos_theta34)

disp('in degrees')
theta34_d = theta34 * rad_deg

%---------------------- Calculate the values using lambart problem
%input r1 r2 ..rv_polar gives the value of phi in 0 to 2pi (quadrent check %included)
r1_p = rv_polar(r1);
r2_p = rv_polar(r2);

phi = r2_p(2) - r1_p(2);

%lets put the value of everything in the problem and lets see how we get %the lamberts problem ...
%then choose the which of the four path u will take
based on tof
format shorte
trajectory1 = zeros(1,5);
    disp('-----------Interplanetary Lambert
Trajectory from EARTH TO VENUS-----------')
[a1,p1,ecc1,itr1,v1_t1,v2_t1] = lambertp(r1,r2,theta12,tof_s(1))
[a_t1,ecc_t1,i_t1,RAAN_t1,argP_t1,f_d]=rv_coe(r1,v1_t1)
disp('Manually correcting true anomaly QUad Check')
f_a_t1=360-f_d
[a_t1,ecc_t1,i_t1,RAAN_t1,argP_t1,f_a]=rv_coe(r2,v2_t1);
disp('Manually correcting true anomaly QUad Check')
f_a_t1=360-f_a

disp('-----------LAUNCH PARAMTERS FROM EARTH-----')
disp('----------Radius of LEO at LAUNCH BE 7000
KM ALTITUDE :~600 KMS---------')
r_launch=7000;
    disp('Velocity in LEO ');
v_leo=sqrt(um_earth/r_launch)
v_leo_m=norm(v_leo);

disp('Velocity to be achieved in interplanetary
traj to venus in HCF')
v_ev_D=v1_t1;
    disp('Velocity to be achieved in
interplanetary traj to venus wrt Earth')
v_inf_D=v_ev_D-v1
    disp('MAGNITUDE of above Velocity to be
achieved in interplanetary traj to venus wrt
Earth')
v_inf_D_m=norm(v_inf_D)

disp('TOTAL VELOCITY THAT SHOULD AT THE POINT
OF BURN TO TRANSFER TO HYPERBOLIC ESCAPE TRAJ FROM
LEO');

V_to_escape=sqrt(2*v_leo_m^2+v_inf_D_m^2)

disp('Total delta V needed to transfer to escape trajectory leading to interplanetary trajectory to Venus');

delta_V_escapte=V_to_escape-v_leo_m

%lambertp=[a,TYPE,p,ecc,itr]
%trajectory_1=[a1,TYPE1,p1,ecc1,itr1,v1_t1,v2_t1]
% this is flyby from venus to mars
    disp('-----------Interplanetary Lambert Trajectory from Venus to MARS---------')
[a2,p2,ecc2,itr2,v1_t2,v2_t2]=lambertp(r2,r3,theta23,tof_s(2))

[a_d,ecc_d,i_d,RAAN_d,argP_d,f_d]=rv_coe(r2,v1_t2)

disp('manually correcting true anomaly in degrees at departure as type 2B')
f_D=360-f_d
[a_d,ecc_d,i_d,RAAN_a,argP_a,f_a]=rv_coe(r3,v2_t2);

f_A=f_a

%trajectory_2=[a,TYPE,p,ecc,itr,v1,v2]
%this is trajectory from mars to pluto
%[a1,a2,a3,a4,a5] =lambertp(r3,r4,theta,tof_s(2))
%trajectory_3=[a1,a2,a3,a4,a5]

%plane_3d(r1,r2,r3)

disp('--------------VENUS FLYBY-----------------

disp('--------------IN HCF-----------------

disp('velocity of arrival in HCF at venus v_sv_A')
v_sv_A= v2_t1

v_sv_A=
disp('velocity of departure in HCF from venus v_sv_D')
v_sv_D=v1_t2

disp('velocity of venus in HCF')
v2

disp('TOTAL delta_v_hcf = v_sv_D-v_sv_A in HCF')
delta_v_hcf = v_sv_D-v_sv_A

disp('TOTAL magnitude delta_v_hcf = v_sv_D-v_sv_A in HCF')
delta_v_hcf_m = norm(delta_v_hcf)

disp('angle change between arrival and departure in HCF')
phi_v_hcf = angle(v_sv_D, v_sv_A) %did v_ add

disp('-------IN VENUS REFERANCE FRAME-------')

disp('velocity of arrival of s/c v wrt venus')
v_inf_A = v_sv_A-v2
v_inf_A_m = norm(v_inf_A)

disp('velocity of departure at infinity from venus of v wrt venus')
v_inf_D = v_sv_D-v2
v_inf_D_m = norm(v_inf_D)

disp('angle change between arrival and departure in venus reference')
phi_inf = angle(v_inf_D, v_inf_A)

disp('TOTAL delta_v_inf = v_inf_D-v_inf_A in venus RF')
delta_v_inf = v_inf_D-v_inf_A

delta_v_inf_m = norm(v_inf_D-v_inf_A)

disp('eccentricity due to phi_inf e=1/sin(phi_inf/2)')
ecc_inf = 1/sin(phi_inf/2)

uv=3.2482e5;

disp('a in hyperbolic orbit due to v_inf_A')
\[ a_{A} = \frac{-uv}{(v_{\text{inf}_A_m})^2} \]

\[
\text{disp('a in hyperbolic orbit due to v_{inf}_D ')}
\]

\[ a_{D} = \frac{-uv}{(v_{\text{inf}_D_m})^2} \]

\[
\text{disp('perigee in hyperbolic orbit due to a_A')}
\]

\[ r_{A}=a_{A}*(1-\text{ecc}_{\infty}) \]

\[
\text{disp('perigee in hyperbolic orbit due to a_D')}
\]

\[ r_{D}=a_{D}*(1-\text{ecc}_{\infty}) \]

\[
\text{disp('perigee in hyperbolic orbit coming less than the radius of VENUS=6051.8kms')}
\]

%------------MARS PLUTO TRAJECTORY------------%

\[
\text{disp('-------------MARS PLUTO Interplanetary TRANSFER TRAJECTORY PARAMETERS-------------')}
\]

\[
\text{format shorte}
\]

\[
[a_{3},p_{3},\text{ecc}_{3},i_{3},v_{1\_t3},v_{2\_t3}]
\]

\[
=\text{lambertp}(r_{3},r_{4},\theta_{34},\text{tof}_s(3))
\]

\[
\text{format shorte}
\]

\[
\text{disp('velocity at arrival at MARS in HCF ')}
\]

\[
v_{\text{sm}_A}=v_{2\_t2};
\]

\[
\text{disp('velocity at departure to pluto from MARS in HCF ')}
\]

\[
v_{\text{sm}_D}=v_{1\_t3} ;
\]

\[
\text{disp('angle change between arrival and departure in HCF at MARS ')}
\]

\[
\phi_{m\_hcf}=\text{angle}(v_{\text{sm}_A},v_{\text{sm}_D})
\]

\[
\text{disp('Total delta v in HCF at MARS ')}
\]

\[
delta_{v\_mars}=v_{\text{sm}_D}-v_{\text{sm}_A};
\]

\[
\text{disp('Total magnitude delta v in HCF at MARS ')}
\]

\[
delta_{v\_mars\_m}= \text{norm}(\delta_{v\_mars})
\]

\[
[a_{3}\_e,\text{ecc}_{3}\_e,\text{i}_{3}\_e,\text{RAAN}_{3}\_e,\text{argP}_{3}\_e,\text{f}_{3\_d}]=\text{rv\_coe}(r_{3},v_{1\_t3})
\]

\[
[a,\text{ecc},i,\text{RAAN, argP},f_{3\_a}]=\text{rv\_coe}(r_{4},v_{2\_t3});
\]

\[
f_{\text{arrival\_pluto}}=f_{3\_a}
\]

\[
\text{disp('-------------MARS FLYBY in MARS REFERNCE FRAME-------------')}
\]
disp('velocity v_inf_m_A arrival v wert mars ') v_inf_m_A=v3-v2_t2 norm(v_inf_m_A) disp('velocity v_inf_m_D departure v wert mars ') v_inf_m_D=v3-v1_t3 norm(v_inf_m_D) disp('angle between v_inf arrival at dep at MARS ') phi_m_inf_mars=angle(v_inf_m_D,v_inf_m_A) disp('Total delta V_inf at MARS ') delta_v_inf_mars=v_inf_m_D-v_inf_m_A disp('Total magnitude of delta V_inf at MARS ') delta_v_inf_mars_m=norm(delta_v_inf_mars) disp('-----------MARS FLYBY HYPERBOLE PARAMETERS-------') ecc_inf_mars=1/sin(phi_m_inf_mars/2) % this needs to be checked for inner and outer angle disp('a in hyperbolic orbit due to v_inf_A ') um_mars=4.281e4; a_A_mars= -um_mars/(norm(v_inf_m_A))^2 disp('a in hyperbolic orbit due to v_inf_D ') a_D_mars= -um_mars/(norm(v_inf_m_D))^2 disp('perigee in hyperbolic orbit at MARS due to a_A') r_A_mars=a_A_mars*(1-ecc_inf_mars) disp('perigee in hyperbolic orbit in MARS due to a_D') r_D_mars=a_D_mars*(1-ecc_inf_mars) disp('55KMS on from mars....radius of mars not included!!')