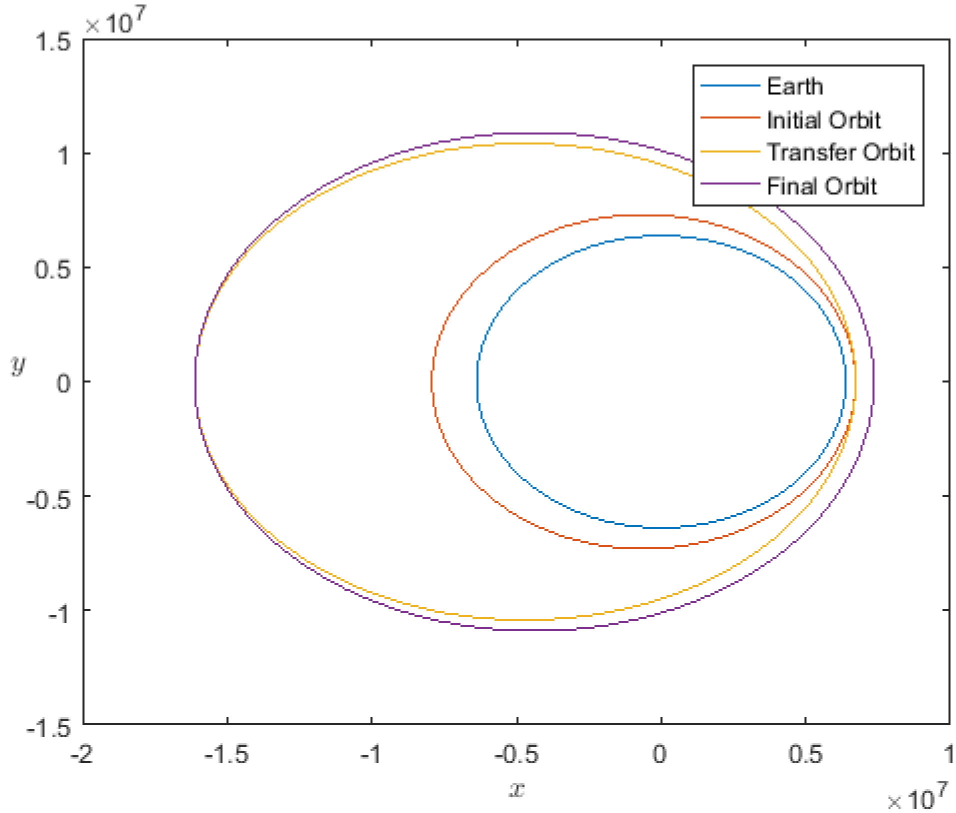
```
% Orbital Mechanics, Project 1
% Shai Sabaroche, 152007282
% Dan Carrollo, 159000852
% Mahmoud Elkayal, 155007835
```

1

```
clear,clc, close all
re=6378e3; mu=3.986e14;
rpi=345e3+re;
rai=1545e3+re;
rpf=975e3+re;
raf=9750e3+re;

% a) Find E, H, Vp, Va, P for the initial orbit
    epsi=(rai-rpi)/(rai+rpi); % eccentricity
    ai=rpi/(1-epsi); % semi major axis
    opi=ai*(1-epsi^2); % orbital parameter
    Ei=-mu/(2*ai); %total specific mechanical energy
    Hi=sqrt(opi*mu); % specific angular momentum
    Vpi=Hi/rpi; % Perigee Velocity
    Vai=Hi/rai; % Apogee Velocity
    Pi=2*pi*sqrt(ai^3/mu); % Orbital Period

% b) Find E, H, Vp, Va, P for the final orbit
    epsf=(raf-rpf)/(raf+rpf); % eccentricity
    af=rpf/(1-epsf); % semi major axis
    opf=af*(1-epsf^2); % orbital parameter
    Ef=-mu/(2*af); %total specific mechanical energy
    Hf=sqrt(opf*mu); % specific angular momentum
    Vpf=Hf/rpf; % Perigee Velocity
    Vaf=Hf/raf; % Apogee Velocity
    Pf=2*pi*sqrt(af^3/mu); % Orbital Period

% c) Find E, H, Vp, Va, P for the transfer trajectory
    at=(rpi+raf)/2;
    Et=-mu/(2*at);
    Vpt=sqrt(2*(Et+mu/rpi));
    Ht=rpi*Vpt;
    Vat=Ht/raf;
    epst=sqrt(1+2*Et*Ht^2/mu^2);

% d) Find the time of flight for the transfer trajectory and delta-Vs
    nut=pi; % true anomaly
    cosut=(epst+cos(nut))/(1+epst*cos(nut)); % eccentric anomaly
    ut=acos(cosut);
    Mt=ut-epst*sin(ut); % mean anomaly
    TOFt=Mt*sqrt(at^3/mu);
    deltaV1=Vpt-Vpi;
    deltaV2=Vaf-Vat;
    deltaVtotal=abs(deltaV1)+abs(deltaV2);
```

```
% e) Plots of trajectories
    theta = 0:pi/50:2*pi;
    re=re.*ones(size(theta));

    xe=re.*cos(theta);
    ye=re.*sin(theta);

    ri=opi./(1+epsi*cos(theta));
    xi=ri.*cos(theta);
    yi=ri.*sin(theta);

    opt=at*(1-epst^2);
    rt=opt./(1+epst*cos(theta));
    xt=rt.*cos(theta);
    yt=rt.*sin(theta);

    rf=opf./(1+epsf*cos(theta));
    xf=rf.*cos(theta);
    yf=rf.*sin(theta);

    figure;
    h1=plot(xe,ye);
    hold on
    h2=plot(xi,yi);
    h3=plot(xt,yt);
    h4=plot(xf,yf);
    legend('Earth', 'Initial Orbit','Transfer Orbit','Final Orbit');
    hold off
    xlabel('$x$','interpreter','latex')
    ylabel('$y
$','interpreter','latex');set(get(gca,'ylabel'),'rotation',0);
    axis([-2e7 1e7 -1.5e7 1.5e7]);

% If a polar representation is required, uncomment the following
 section
%     figure;
%     h1=polar(theta,rf);
%     hold on
%     h2=polar(theta,ri);
%     h3=polar(theta,rt);
%     h4=polar(theta,re);
%
%     hHiddenText = findall(gca,'type','text');
%     Angles = 0 : 30 : 330;
%     hObjToDelete = zeros( length(Angles)-4, 1 );
%     k = 0;
%
% for ang = Angles
%    hObj = findall(hHiddenText,'string',num2str(ang));
%    switch ang
%    case 0
%       set(hObj,'string','0','HorizontalAlignment','Left');
%    case 90
%       set(hObj,'string','3\pi/2','VerticalAlignment','Bottom');
```

```
%    case 180
%       set(hObj,'string','\pi','HorizontalAlignment','Right');
%    case 270
%       set(hObj,'string','\pi/2','VerticalAlignment','Top');
%    otherwise
%       k = k + 1;
%       hObjToDelete(k) = hObj;
%    end
% end
% delete( hObjToDelete(hObjToDelete~=0) );
%  title('Orbital Transfer for Problem 1');
%
%     hold off
```



2

a)

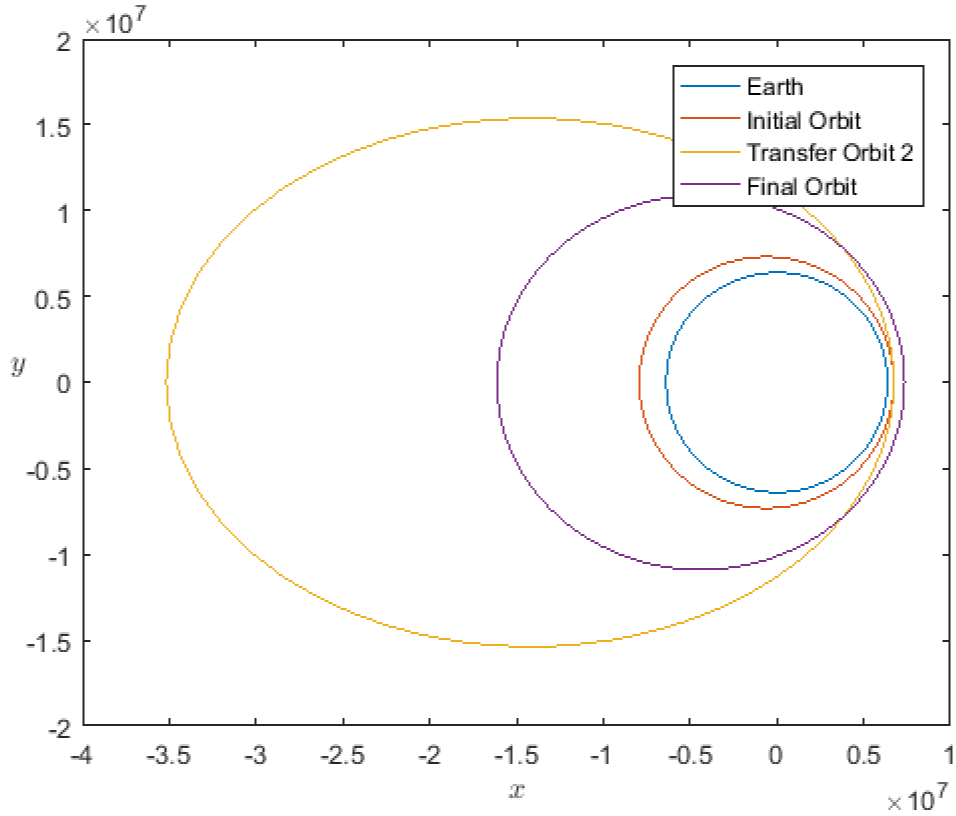
```
close all
rpt2=rpi;
epst2=1.65*epst;
at2=rpt2/(1-epst2);
Et2=-mu/(2*at2);
Vpt2=sqrt(2*(Et2+mu/rpt2));
Ht2=rpi*Vpt2;
```

```
% b)
  theta = 0:pi/50:2*pi;
  re=re.*ones(size(theta));

  xe=re.*cos(theta);
  ye=re.*sin(theta);

  ri=opi./(1+epsi*cos(theta));
  xi=ri.*cos(theta);
  yi=ri.*sin(theta);

  opt2=at2*(1-epst2^2);
  rt2=opt2./(1+epst2*cos(theta));
  xt2=rt2.*cos(theta);
  yt2=rt2.*sin(theta);

  rf=opf./(1+epsf*cos(theta));
  xf=rf.*cos(theta);
  yf=rf.*sin(theta);

figure;
  h1=plot(xe,ye);
  hold on
  h2=plot(xi,yi);
  h3=plot(xt2,yt2);
  h4=plot(xf,yf);
  legend('Earth', 'Initial Orbit','Transfer Orbit 2','Final Orbit');
    xlabel('$x$','interpreter','latex')
  ylabel('$y
$','interpreter','latex');set(get(gca,'ylabel'),'rotation',0);

  hold off

  L1=[xf;yf]; %computed using the attached m file "InterX' which has
all the code
  L2=[xt2;yt2];
  P=InterX(L1,L2);
  Xinter=P(1);
  Yinter=P(2);
  Rinter=sqrt(Xinter^2+Yinter^2) % intersection distance from centre
of earth
  thetaInter=atan(Yinter/Xinter) % true anomaly at intersection

% d) Elevation Angle and Velocity of final orbit at intersection
    phifinter=atan(epsf*sin(thetaInter)/(1+epsf*cos(thetaInter)))
    Vinterf=sqrt(2*(Ef+mu/Rinter))

% e) Elevation angle and velocity of transfer orbit at intersection
    phitinter=atan(epst2*sin(thetaInter)/(1+epst2*cos(thetaInter)))
    Vintert=sqrt(2*(Et2+mu/Rinter))

% f)
    deltaV1fort2=Vpt2-Vpi % initial impulsive thrust from initial
orbit to transfer 2
```

```
    alpha=phitinter-phifinter %angle between the interception angles

 deltaV2fort2=sqrt(Vinterf^2+Vintert^2-2*Vinterf*Vintert*cos(alpha)) %
 final orbit insertion
    deltaVtotal2=abs(deltaV1fort2)+abs(deltaV2fort2)
```

```
Warning: NARGCHK will be removed in a future release. Use NARGINCHK or
NARGOUTCHK instead.

Rinter =

   8.6433e+06


thetaInter =

    1.1041


phifinter =

    0.2783


Vinterf =

   7.6343e+03


phitinter =

    0.4349


Vintert =

   8.5562e+03


deltaV1fort2 =

   1.9684e+03


alpha =

    0.1566


deltaV2fort2 =

   1.5646e+03
```

```
deltaVtotal2 =

   3.5331e+03
```



Published with MATLAB® R2016a