Advanced Topics in Astrodynamics and Trajectory Design Pre-Course Material

The following tutorial should be completed before the first session of the Advanced Topics in Astrodynamics and Trajectory Design (ATATD). The purpose of the exercise is to ensure that your Matlab skills are sufficiently advanced to ensure a good progress during the ATATD tutorials.

Please, complete the following exercise using Matlab.

Exercise. Solve the numerical integration of the differential equations of the two-body motion:

$$\ddot{\mathbf{r}} + \frac{\mu_E}{r^3}\mathbf{r} = 0 \tag{1}$$

For the following initial conditions:

$$\mathbf{X}(t=0) = [\text{-}18{,}676 \quad 6{,}246 \quad 12{,}474 \quad 0.551 \quad \text{-}1.946 \quad \text{-}3.886]$$

Where the three first components correspond to a position vector $\mathbf{r}$ and the three last components to a velocity vector $\dot{\mathbf{r}}$ in units of km and km/s, respectively, and μ_E is the gravitational constant of the Earth.

Plot the results in Matlab with an adequate representation, such that would allow you to recognize the orbit.

Notes and Hints. Equation (1) is a second order differential equation describing the so-called restricted two-body motion. An ordinary differential equation, often referred simply as ODE, is an equality involving a function and its derivatives. Equation (1) is referred as a second order, since its highest derivative involve a 2nd derivative of the position vector $\mathbf{r}$ with respect to time (i.e. the acceleration).

Matlab provides a series of functions and tools to solve the numerical integration of ODEs. Please, take your time revising Matlab tutorials if you are not familiar with these tools.

https://uk.mathworks.com/help/matlab/ordinary-differential-equations.html

Solution. In order to solve Equation (1), we will need first to describe the equation (1) as a system of equations in its first order form:

$$\begin{cases} \ddot{x} + \dfrac{\mu_E}{r^3}x = 0 \\ \ddot{y} + \dfrac{\mu_E}{r^3}y = 0 \\ \ddot{z} + \dfrac{\mu_E}{r^3}z = 0 \end{cases} \Longrightarrow \begin{cases} \dot{x} = v_x \\ \dot{y} = v_y \\ \dot{z} = v_z \\ \dot{v}_x = -\dfrac{\mu_E}{r^3}x \\ \dot{v}_y = -\dfrac{\mu_E}{r^3}y \\ \dot{v}_z = -\dfrac{\mu_E}{r^3}z \end{cases} \Longrightarrow \begin{pmatrix} x & y & z & v_x & v_y & v_z \end{pmatrix}$$

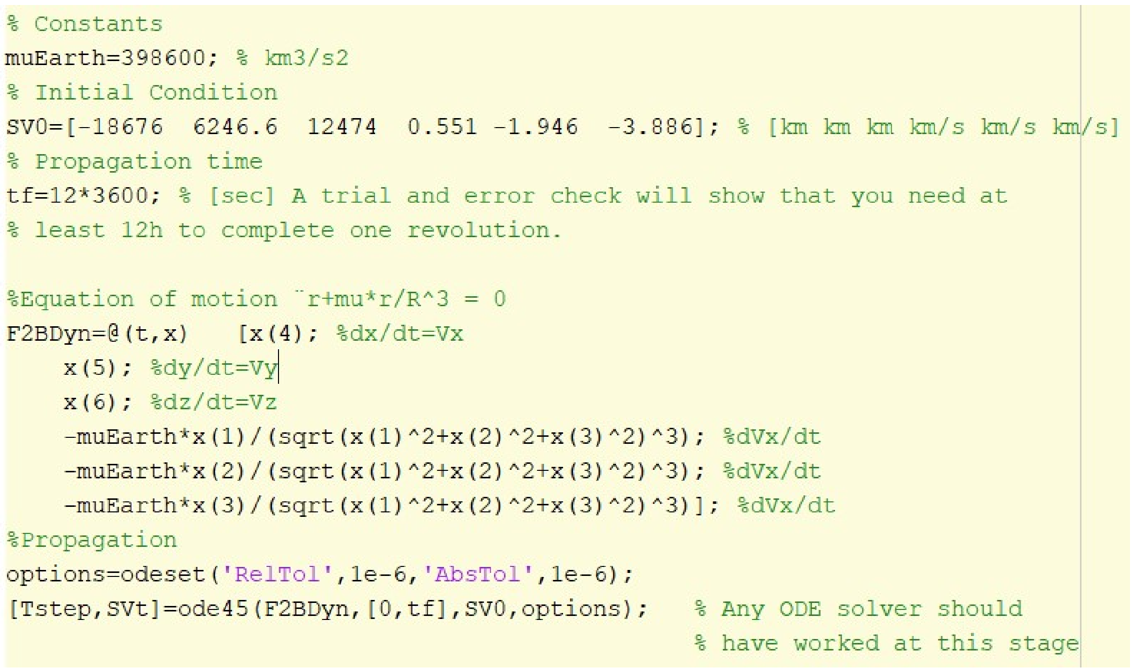
Where the system of equations that need to be solved is:

$$\dot{x} = v_x$$
$$\dot{y} = v_y$$
$$\dot{z} = v_z$$
$$\dot{v}_x = -\frac{\mu_E}{r^3}x$$
$$\dot{v}_y = -\frac{\mu_E}{r^3}y$$
$$\dot{v}_z = -\frac{\mu_E}{r^3}z$$

Hence, the following system need to be written as a Matlab function. Possibly, the best option is to do it as an *anonymous function*. If you are not aware of how to do this, please take your time revising the following tutorial:

http://uk.mathworks.com/help/matlab/matlab_prog/anonymous-functions.html

After revising how to write anonymous functions and how to solve ODEs, your code should look something like:



```
% Constants
muEarth=398600; % km3/s2
% Initial Condition
SV0=[-18676  6246.6  12474  0.551 -1.946  -3.886]; % [km km km km/s km/s km/s]
% Propagation time
tf=12*3600; % [sec] A trial and error check will show that you need at
% least 12h to complete one revolution.

%Equation of motion ¨r+mu*r/R^3 = 0
F2BDyn=@(t,x)    [x(4); %dx/dt=Vx
    x(5); %dy/dt=Vy
    x(6); %dz/dt=Vz
    -muEarth*x(1)/(sqrt(x(1)^2+x(2)^2+x(3)^2)^3); %dVx/dt
    -muEarth*x(2)/(sqrt(x(1)^2+x(2)^2+x(3)^2)^3); %dVx/dt
    -muEarth*x(3)/(sqrt(x(1)^2+x(2)^2+x(3)^2)^3)]; %dVx/dt
%Propagation
options=odeset('RelTol',1e-6,'AbsTol',1e-6);
[Tstep,SVt]=ode45(F2BDyn,[0,tf],SV0,options);   % Any ODE solver should
                                                % have worked at this stage
```

Matrix SVt will now contain the state vector at a corresponding grid of times Tstep. This can now be plotted so that the orbit can be easily recognized.

You may be tempted to simply write the following command:

```
figure
plot3(SVt(:,1),SVt(:,2),SVt(:,3))
```

While the above lines may often work, Matlab automatically autoscales figures so that the above command will give you the impression of a circular orbit.

Try instead the following:

```
Re=6378; %km
figure
axis equal
hold on
[X,Y,Z] = sphere(20);
X=X*Re; Y=Y*Re; Z=Z*Re;
surf(X,Y,Z)
plot3(SVt(:,1),SVt(:,2),SVt(:,3))
view(75,20)
```

The plot now should provide you a pretty good guess of the type of orbit you are looking at.