AE 502 Project 2

Linyi Hou | linyih2 | github.com/TigerHou2

March 22, 2021

Problem 1

Problem Statement: Assume an Earth-relative orbit is given by the initial orbit elements $a = 7000$ km, $e = 0.05$, $i = 45^o$, $\Omega = 0^o$, $\omega = 45^o$, $M_0 = 0^o$. Assume the disturbance acceleration is solely due to the J_2 gravitational acceleration given below, where $J_2 = 1082.63 \times 10^{-6}$ and $r_{eq} = 6378.137$ km.

$$\mathbf{a}_{J_2} = -\frac{3}{2} J_2 \left(\frac{\mu}{r^2}\right)\left(\frac{r_{eq}}{r}^2\right) \begin{bmatrix} \left(1 - 5\left(\frac{z}{r}\right)^2\right)\frac{x}{r} \\ \left(1 - 5\left(\frac{z}{r}\right)^2\right)\frac{y}{r} \\ \left(3 - 5\left(\frac{z}{r}\right)^2\right)\frac{z}{r} \end{bmatrix} \tag{1}$$

Part a

Q: Using Cowell's method, set up a numerical simulation to solve for $\{\mathbf{x}(t), \dot{\mathbf{x}}(t)\}$ over 10 orbit periods.

A: Cowell's method refers to the direct integration of the equations of motion including perturbation. Therefore, the following equations of motion (EOMs) apply:

$$\dot{\mathbf{r}} = \mathbf{v} \tag{2}$$

$$\ddot{\mathbf{r}} = -\frac{\mu}{r^3}\mathbf{r} + \mathbf{a}_{J_2} \tag{3}$$

The above EOMs were propagated for 10 orbit periods using the MATLAB `ode45` function, and the results are shown in Fig. 1 below.

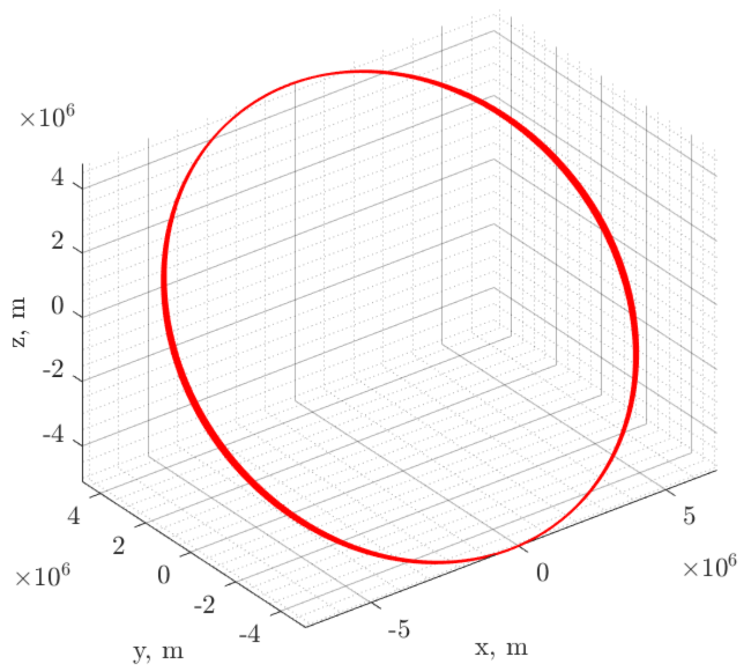

Figure 1: Orbit propagation results using Cowell's method.

Part b

Q: Translate the $\{\mathbf{x}(t), \dot{\mathbf{x}}(t)\}$ coordinates into the corresponding classical orbit elements $\{a, e, i, \Omega, \omega, M\}$.

A: The equations used to convert $\{\mathbf{x}, \dot{\mathbf{x}}\}$ into the six orbit elements are shown below:

$$a = \frac{r}{2 - rv^2/\mu} \tag{4}$$

$$\mathbf{e} = \frac{1}{\mu}\left[\left(v^2 - \frac{\mu}{r}\right)\mathbf{r} - \left(\mathbf{r}^T\mathbf{v}\right)\mathbf{v}\right] \tag{5}$$

$$i = \arccos\left(\frac{\mathbf{h}^T\hat{\mathbf{K}}}{h}\right) \tag{6}$$

$$\Omega = \arccos\left(\frac{\mathbf{n}^T\hat{\mathbf{I}}}{n}\right) \tag{7}$$

$$\omega = \arccos\left(\frac{\mathbf{n}^T\mathbf{e}}{ne}\right) \tag{8}$$

$$M = E - e\sin E \tag{9}$$

where the variables $\mathbf{h}$, $\mathbf{n}$, $\hat{\mathbf{I}}$, $\hat{\mathbf{K}}$, and E are defined as follows:

$$E = 2\arctan\left(\sqrt{\frac{1-e}{1+e}}\tan\frac{f}{2}\right) \tag{10}$$

$$f = \arccos\left(\frac{\mathbf{e}^T\mathbf{r}}{er}\right) \tag{11}$$

$$\hat{\mathbf{I}} = [1,0,0]^T \ , \ \hat{\mathbf{K}} = [0,0,1]^T \tag{12}$$

$$\mathbf{h} = \mathbf{r}\times\mathbf{v} \ , \ \mathbf{n} = \hat{\mathbf{K}}\times\frac{\mathbf{h}}{h} \tag{13}$$

The six classical orbit elements are shown in Fig. 2. Note that while RAAN technically spans $[0, 2\pi)$, the range $[-\pi, \pi)$ was chosen instead to better visualize the data.

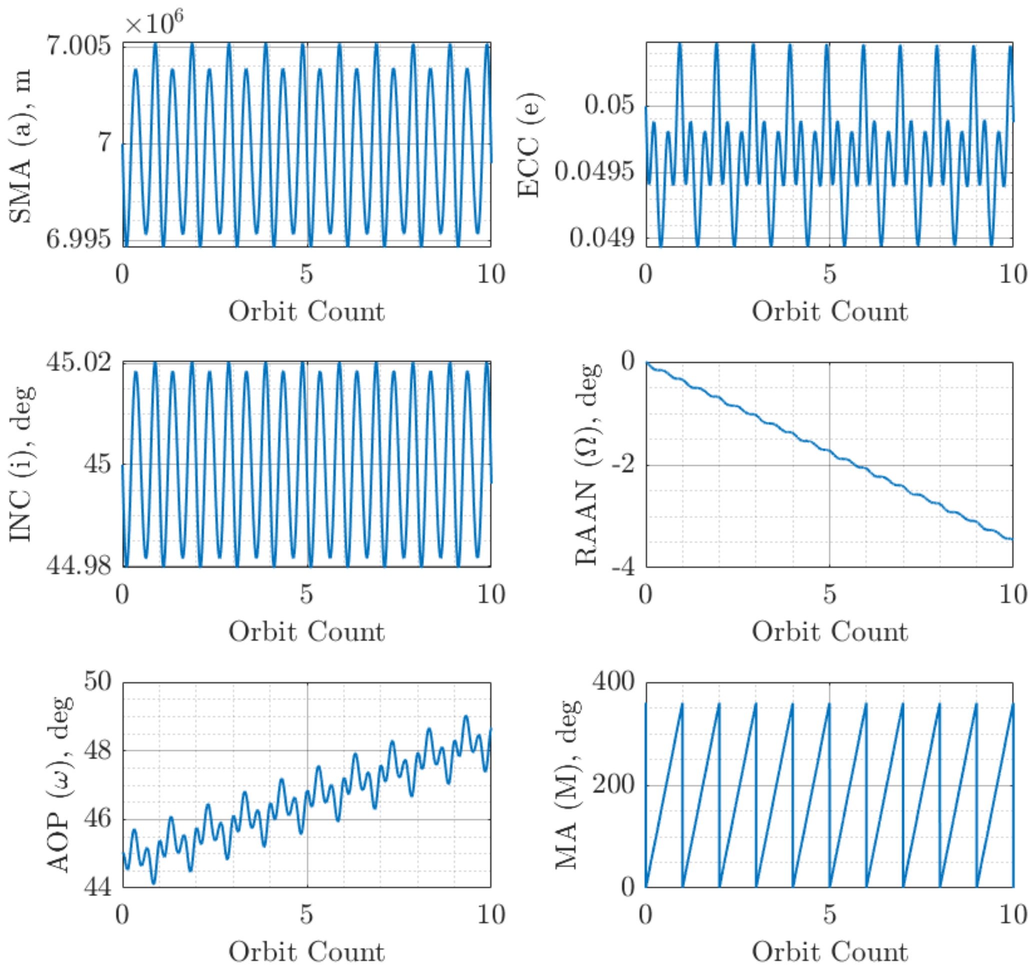

Figure 2: Orbit elements over 10 orbit periods.

The orbit's nodal precession due to J2 perturbations can be clearly seen by the steady decrease in Ω over the course of 10 orbits. Perigee rotation can also be observed from the increase in ω. Other orbit elements also exhibited oscillatory behavior, but it appears that their mean/nominal values did not deviate noticeably over time. Note that the change in mean anomaly is linear since it is plotted against the orbit count, both of which are proportional to time.

Part c

Q: Compare the numerically computed motion of the elements in (b) to the average orbit variations (use the mean element rate equations).

A: The mean element rate equations are shown below, and the difference between the numerically simulated parameters and the mean element rate propagation results are plotted in Fig. 3.

$$\frac{da}{dt} = 0 \ ; \ \frac{de}{dt} = 0 \ ; \ \frac{di}{dt} = 0 \tag{14}$$

$$\frac{d\Omega}{dt} = -\frac{3}{2} J_2 n \left(\frac{r_{eq}}{p}\right)^2 \cos i \tag{15}$$

$$\frac{d\omega}{dt} = \frac{3}{4} J_2 n \left(\frac{r_{eq}}{p}\right)^2 (5cos^2 i - 1) \tag{16}$$

$$\frac{dM}{dt} = \frac{3}{4} J_2 n \left(\frac{r_{eq}}{p}\right)^2 \sqrt{1-e^2}\,(3cos^2 i - 1) \tag{17}$$

where p is the semi-latus rectum and n is the mean motion:

$$p = a(1-e^2) \tag{18}$$

$$n = \sqrt{\frac{\mu}{a^3}} \tag{19}$$

The mean element rates reflect the averaged behavior of the orbit elements over an integer number of orbit periods, and thus are not able to characterize the variations of any element within one orbit period. As such, deviations arise within a single orbit period.

Nonetheless, the mean element rates are able to match the values of the numerical integration results at the beginning of each period, which is clearly illustrated by the periodic zero-intercepts in each subplot.

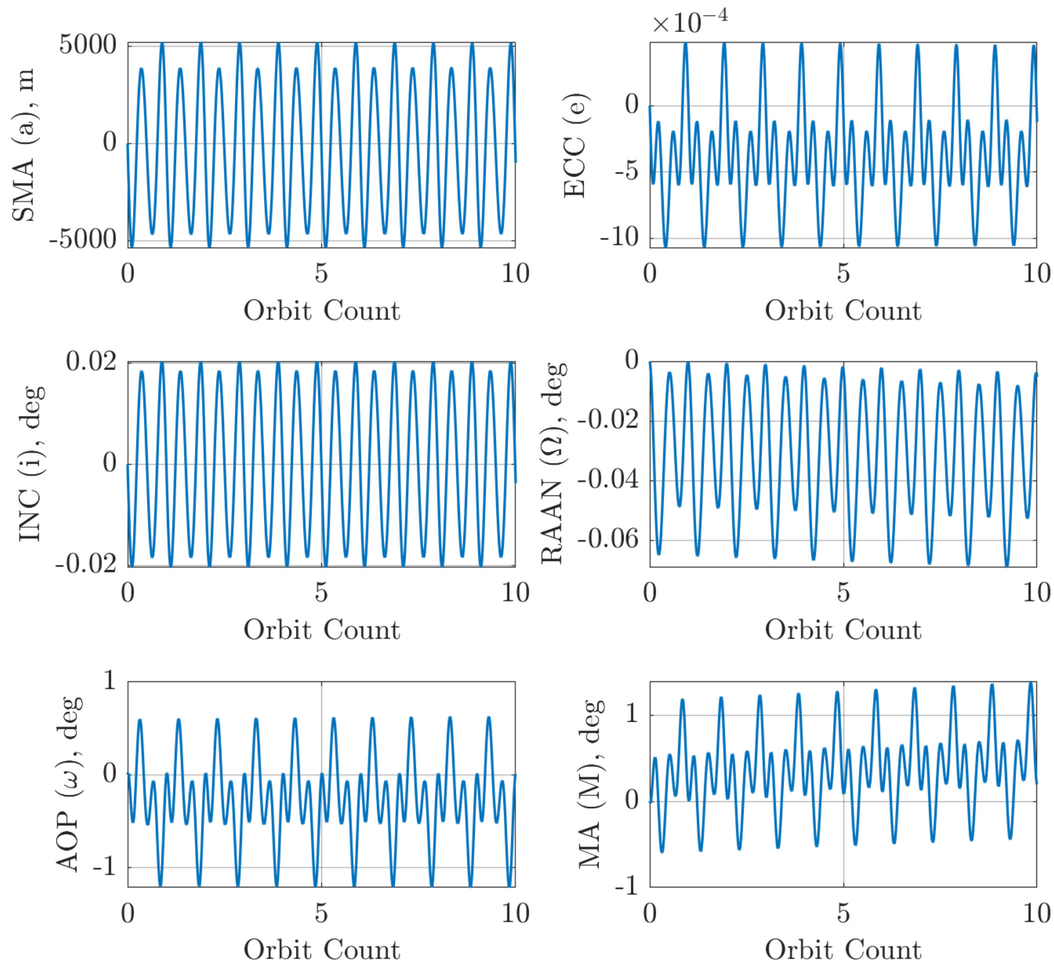

Figure 3: Difference between the numerical integration and mean element rate simulation results for the orbit elements.

Problem 2

Problem Statement: Assume the same initial conditions and force model in Problem 1.

Part a

Q: Use Gauss' variational equations to propagate the initial conditions for 10 orbit periods.

A: Gauss' variational equations is based upon the variation of parameters with a coordinate frame defined as follows:

$$\hat{\mathbf{i}}_r = \frac{\mathbf{r}}{r} \tag{20}$$
$$\hat{\mathbf{i}}_h = \frac{\mathbf{h}}{h} \tag{21}$$
$$\hat{\mathbf{i}}_\theta = \hat{\mathbf{i}}_h \times \hat{\mathbf{i}}_r \tag{22}$$

The variation in the orbit elements were provided in the lecture notes, as follows:

$$\frac{da}{dt} = \frac{1}{h}2a^2\left(e \sin f \delta_r + \frac{p}{r}\delta_\theta\right) \tag{23}$$
$$\frac{de}{dt} = \frac{1}{h}\left(p \sin f \delta_r + \left((p + r) \cos f + re\right)\delta_\theta\right) \tag{24}$$
$$\frac{di}{dt} = \frac{1}{h} r \cos \theta \delta_h \tag{25}$$
$$\frac{d\Omega}{dt} = \frac{1}{h}\frac{r \sin \theta}{\sin i}\delta_h \tag{26}$$
$$\frac{d\omega}{dt} = \frac{1}{he}\left(-p \cos f \delta_r + (p + r) \sin f \delta_\theta\right) - \frac{r \sin \theta \cos i}{h \sin i}\delta_h \tag{27}$$
$$\frac{dM}{dt} = n + \frac{b}{ahe}\left((p \cos f - 2re)\delta_r - (p + r) \sin f \delta_\theta\right) \tag{28}$$

The results are shown in Fig. 4.

Part b

Q: Use the MATLAB functions `tic` and `toc` to compute the computation time to propagate the orbit in Problem 1(a) and Problem 2(a). Take the average of 10 runs.

A: The run time required for each propagation method is tabulated in Table 1. The average time required for direct numerical integration is 95.56 milliseconds, while the average time required for Gauss' variational method is 35.53 milliseconds.

Table 1: Propagation time (ms) for direct numerical integration and Gauss' variational method over 10 simulations.

Run #	1	2	3	4	5	6	7	8	9	10
Numerical	95.72	96.54	95.12	98.75	92.42	97.78	91.54	97.92	92.75	101.01
Gauss	29.74	31.26	52.75	33.38	37.89	35.78	32.89	36.51	33.00	32.05

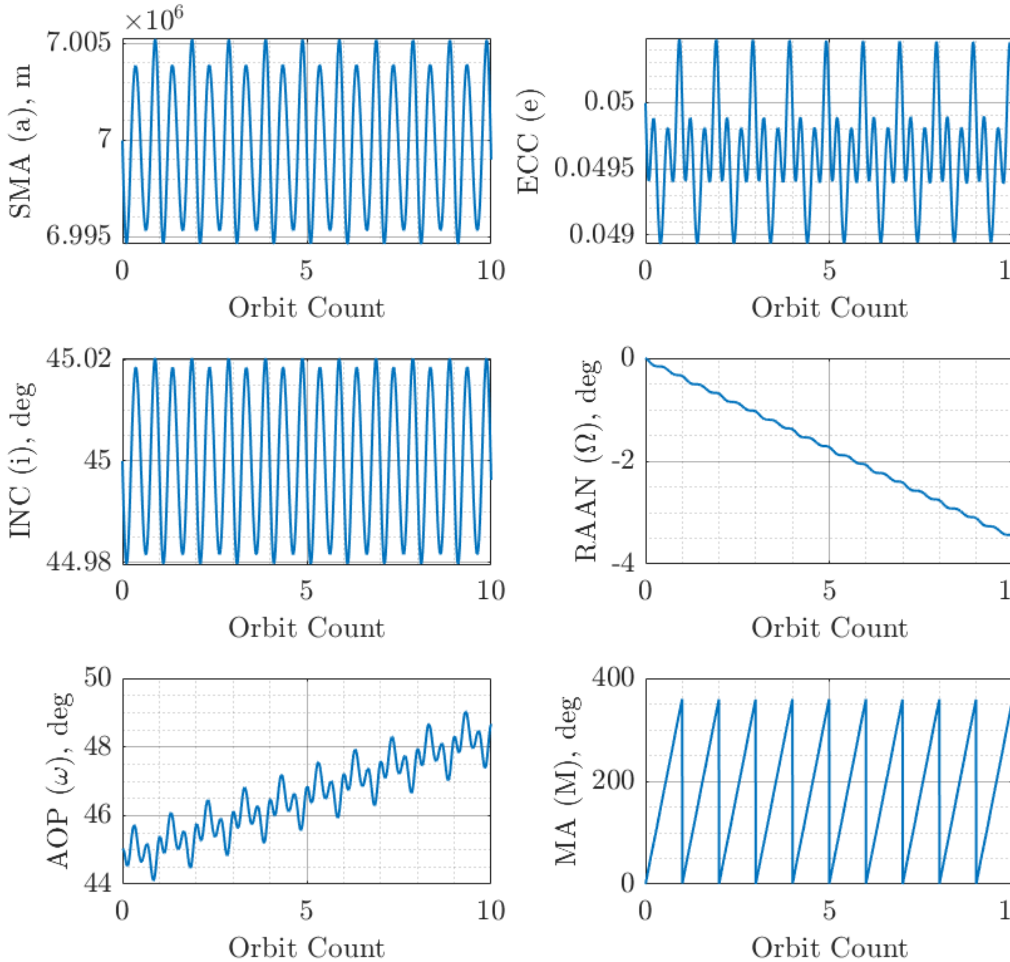

Figure 4: Orbit elements propagated over 10 orbit periods using Gauss' variational method.

Problem 3

Problem Statement: Assume the same initial conditions in Problem 1, but also include atmospheric drag into the force model (use Vallado's *Exponential Drag Model*). Repeat Problems 1(a) and 1(b), and compare the result to the values of $\{a, e, i, \Omega, \omega, M\}$ obtained in Problem 1(b).

A: The perturbation due to drag is calculated by the equation

$$\mathbf{a}_{drag} = -\frac{1}{2}\frac{C_D A}{m}\rho v^2 \frac{\mathbf{v}}{v} \tag{29}$$

where ρ can be modeled using the exponential drag model:

$$\rho = \rho_0 \exp\left[-\frac{h_{ellp} - h_0}{H}\right] \tag{30}$$

where ρ_0 is the reference density, h_0 is the reference altitude, and h_{ellp} is the actual altitude above the ellipsoid (of the central body). From Vallado, the relevant sections of the exponential atmosphere model are tabulated below:

Table 2: Exponential atmosphere model for the Earth.

h_{ellp} (km)	h_0 (km)	ρ_0 (kg/m^3)	H (km)
200	250	2.789e-10	37.105
250	300	7.248e-11	45.546
300	350	2.418e-11	53.628
350	400	9.518e-12	53.298
400	450	3.725e-12	58.515
450	500	1.585e-12	60.828
500	600	6.967e-13	63.822
600	700	1.454e-13	71.835
700	800	3.614e-14	88.667
800	900	1.170e-14	124.64
900	1000	5.245e-15	181.05

The same process as outlined in Problem 1 were applied to generate the classical orbit elements with the inclusion of atmospheric drag. The difference between the simulation results with atmospheric drag and simulation results without drag is shown in Fig. 5.

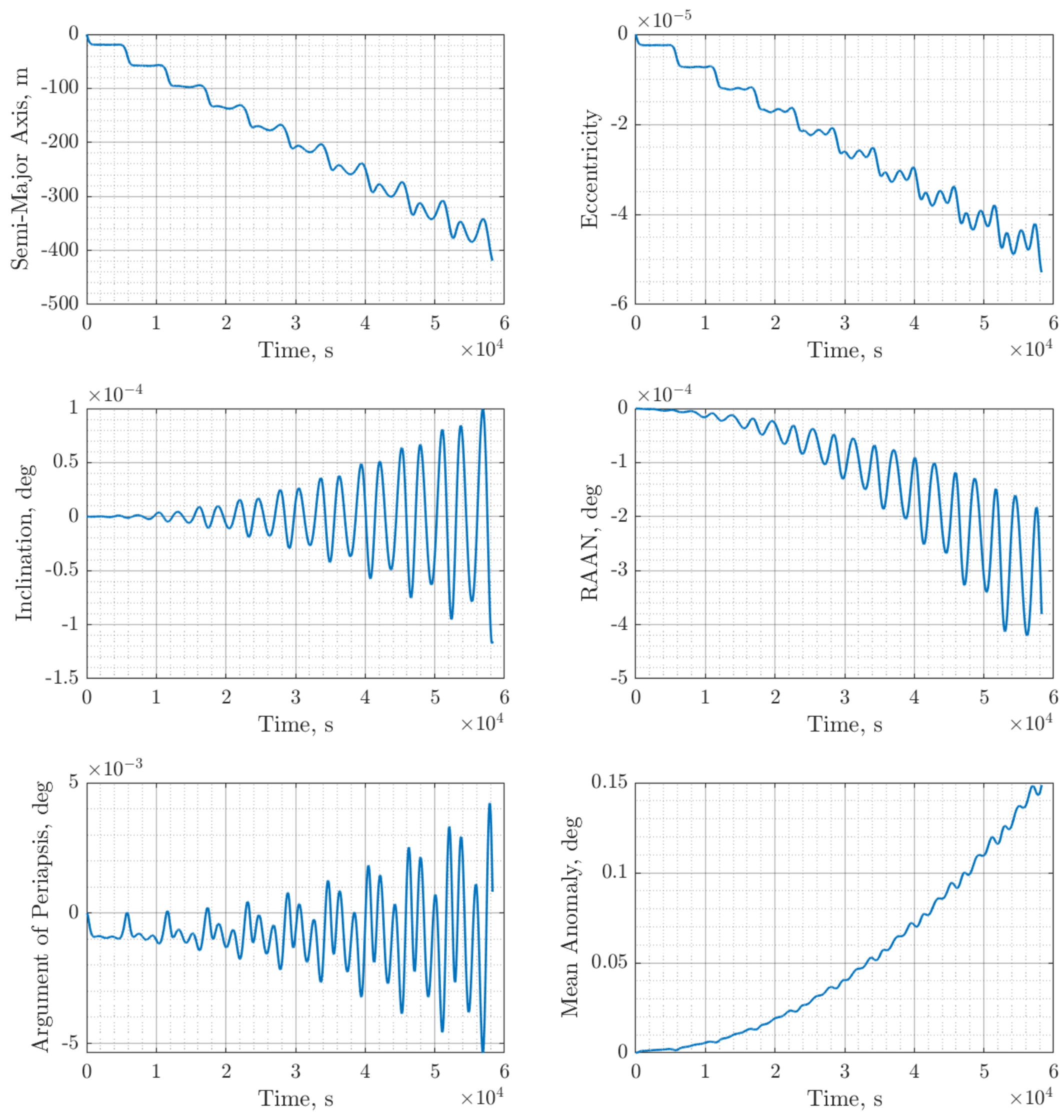

Figure 5: Difference between orbital elements simulated with drag and without drag, over 10 orbit periods using Cowell's method.

Appendix A: Code for Problem 1

```

%% AE 502 HW2 Problem 1, Spring 2021
%   Tiger Hou
close all
clear;clc

%% Part a

tic

% Earth orbit general parameters
mu = 3.986e14; % m^3/s^2, Earth gravitational parameter
J2 = 1082.63e-6; % J2 perturbation coefficient
req = 6378.137e3; % km, equatorial radius

% initial conditions
a = 7000e3;
e = 0.05;
i = deg2rad(45);
o = deg2rad(0);
w = deg2rad(45);
M0 = deg2rad(0);
E0 = kepler(M0,e);
f0 = 2 * atan(sqrt((1+e)/(1-e))*tan(E0/2));

% find initial position and velocity vectorsr0
[r0,v0] = Get_Orb_Vects([a,e,i,o,w,f0],mu);

% define the perturbation equation as p(rv)
p = @(rv) -3/2 * J2 * (mu/norm(rv)^2) * (req/norm(rv))^2 * ...
    [ ( 1-5*(rv(3)/norm(rv))^2 ) * rv(1)/norm(rv); ...
      ( 1-5*(rv(3)/norm(rv))^2 ) * rv(2)/norm(rv); ...
      ( 3-5*(rv(3)/norm(rv))^2 ) * rv(3)/norm(rv) ];

% calculate orbit period
T = 2*pi * sqrt(a^3/mu); % seconds

% ode45
```

```
rv0 = [r0;v0];
options = odeset('RelTol',1e-9,'AbsTol',1e-12);
[tOut,rvOut] = ode45(@(t,rv)ff(rv,mu,1,p),[0,10*T],rv0,options);

toc

% plotting
rvOut = rvOut';
figure(1)
plot3(rvOut(1,:), rvOut(2,:),rvOut(3,:),'r','LineWidth',1.2)
xlabel('x, m')
ylabel('y, m')
zlabel('z, m')
axis equal
setgrid
latexify(16,14,14)

%% Part b
% convert position, velcoity data into orbit parameters
N = size(rvOut,2);
paramOut = nan(size(rvOut));
for j = 1:N
    [a_,e_,i_,o_,w_,f_] = Get_Orb_Params(rvOut(1:3,j),rvOut(4:6,j),mu);
    E_ = 2 * atan(sqrt((1-norm(e_))/(1+norm(e_)))*tan(f_/2));
    M_ = E_ - norm(e_)*sin(E_);
    paramOut(:,j) = [a_,norm(e_),i_,o_,w_,M_]';
end

% plot results
ylabel_vec = {'SMA (a), m', ...
              'ECC (e)', ...
              'INC (i), deg', ...
              'RAAN ($\Omega$), deg', ...
              'AOP ($\omega$), deg', ...
              'MA (M), deg'};
xlabel_val = 'Orbit Count';
figure(2)
for j = 1:6
    subplot(3,2,j)
    dat = paramOut(j,:);
```

```
    if j == 4 % RAAN loop-around after 2*pi
        dat = mod(dat+pi,2*pi)-pi;
    end
    if j == 6 % mean anomaly loop-around after 2*pi
        dat = mod(dat,2*pi);
    end
    if j >= 3 % conversion to degrees
        dat = rad2deg(dat);
    end
    plot(tOut/T,dat,'Linewidth',1.2)
    xlabel(xlabel_val)
    ylabel(ylabel_vec{j})
    setgrid
end
latexify(20,18,14)

%% Part c
n = sqrt(mu/a^3); % mean motion
p = a*(1-e^2); % semi-latus rectum
dadt = 0;
dedt = 0;
didt = 0;
dodt = -3/2*J2*n*(req/p)^2*cos(i);
dwdt = 3/4*J2*n*(req/p)^2*(5*cos(i)^2-1);
dMdt = 3/4*J2*n*(req/p)^2*sqrt(1-e^2)*(3*cos(i)^2-1);

param0 = [a,e,i,o,w,M0]';

paramMean = ([dadt,dedt,didt,dodt,dwdt,dMdt]') * (tOut') + param0;

for j = 1:6
    subplot(3,2,j)
    dat = paramMean(j,:);
    if j == 4 % RAAN loop-around after 2*pi
        dat = mod(dat+pi,2*pi)-pi;
    end
    if j == 6 % mean anomaly loop-around after 2*pi
        dat = mod(dat,2*pi);
    end
    if j >= 3 % conversion to degrees
```

```
        dat = rad2deg(dat);
    end
    hold on
    plot(tOut/T,dat,'r','Linewidth',1.2)
    setgrid
    grid minor
    hold off
%     xlabel(xlabel_val)
%     ylabel(ylabel_vec{j})
    if j == 4 % add legend in the relatively empty plot
        legend('Numerical','Mean Motion','Location','best')
    end
end
latexify(20,18,14)

% figure(2)
% plot(tOut,paramMean(6,:))
% hold on
% plot(tOut,paramOut(6,:))
% hold off

%% function definitions
function rv_dot = ff(rv,mu,N,p)
% takes the 6xN position & velocity vector and computes the derivative
%   where N is the number of particles to track
%   also applies perturbation in the form of a function handle p
%   which takes argument p(r)

rv_dot = zeros(6,N);

for i = 1:N
    % velocity
    rv_dot(1:3,i) = rv(4:6,i);
    % acceleration
    rv_dot(4:6,i) = -mu/norm(rv(1:3,i))^3*rv(1:3,i) + p(rv(1:3,i));
end

end
```

Appendix B: Code for Problem 2

```

%% AE 502 HW2 Problem 2, Spring 2021
%   Tiger Hou
close all
clear;clc

tic

mu = 3.986e14; % m^3/s^2, Earth gravitational parameter
J2 = 1082.63e-6; % J2 perturbation coefficient
req = 6378.137e3; % km, equatorial radius

% initial conditions
a = 7000e3;
e = 0.05;
i = deg2rad(45);
o = deg2rad(0);
w = deg2rad(45);
M0 = deg2rad(0);
E0 = kepler(M0,e);
f0 = 2 * atan(sqrt((1+e)/(1-e))*tan(E0/2));

% define the perturbation equation in the inertial frame as p(r)
p = @(r) -3/2 * J2 * (mu/norm(r)^2) * (req/norm(r))^2 * ...
    [ ( 1-5*(r(3)/norm(r))^2 ) * r(1)/norm(r); ...
      ( 1-5*(r(3)/norm(r))^2 ) * r(2)/norm(r); ...
      ( 3-5*(r(3)/norm(r))^2 ) * r(3)/norm(r) ];

% calculate orbit period
T = 2*pi * sqrt(a^3/mu); % seconds

% ode45
param0 = [a,e,i,o,w,M0]';
options = odeset('RelTol',1e-9,'AbsTol',1e-12);
[tOut,paramOut] = ode45(@(t,params)ff(params,mu,p),[0,10*T],param0,
    options);

toc
```

```matlab

% plotting
paramOut = paramOut';
ylabel_vec = {'SMA (a), m', ...
              'ECC (e)', ...
              'INC (i), deg', ...
              'RAAN ($\Omega$), deg', ...
              'AOP ($\omega$), deg', ...
              'MA (M), deg'};
xlabel_val = 'Orbit Count';
figure(2)
for j = 1:6
    subplot(3,2,j)
    dat = paramOut(j,:);
    if j == 4 % RAAN loop-around after 2*pi
        dat = mod(dat+pi,2*pi)-pi;
    end
    if j == 6 % mean anomaly loop-around after 2*pi
        dat = mod(dat,2*pi);
    end
    if j >= 3 % conversion to degrees
        dat = rad2deg(dat);
    end
    plot(tOut/T,dat,'Linewidth',1.2)
    xlabel(xlabel_val)
    ylabel(ylabel_vec{j})
    setgrid
end
latexify(20,18,14)

%% function definitions
function param_dot = ff(params,mu,A)
% takes the 6x1 orbit parameters and computes the derivative using Gauss'
% variation of parameters
%   the orbit parameters are a, e, i, o, w, M
%   also applies perturbation in the form of a function handle A
%   which takes argument A(params)

a = params(1);
e = params(2);
```

```
i = params(3);
o = params(4);
w = params(5);
M = params(6);
E = kepler(M,e);
f = 2 * atan(sqrt((1+e)/(1-e))*tan(E/2));
p = a*(1-e^2); % semi-latus rectum
n = sqrt(mu/a^3); % mean motion
b = sqrt(a*p);

[R,V] = Get_Orb_Vects([a,e,i,o,w,f],mu);
r = norm(R);
H = cross(R,V);
h = norm(H);

% find the LVLH reference frame basis vectors
ir = R / r;
in = H / h;
it = cross(in,ir);

aR = A(R)' * ir; % radial perturbation
aT = A(R)' * it; % theta perturbation
aN = A(R)' * in; % normal perturbation

dadt = (2*a^2/h) * ( e*sin(f)*aR + p/r*aT );
dedt = 1/h * ( p*sin(f)*aR + ((p+r)*cos(f)+r*e)*aT );
didt = 1/h * r*cos(w+f) * aN;
dodt = (r*sin(w+f)) / (h*sin(i)) * aN;
dwdt = 1/h/e * ( -p*cos(f)*aR + (p+r)*sin(f)*aT ) ...
        - (r*sin(w+f)*cos(i)) / (h*sin(i)) * aN;
dmdt = n + b/(a*h*e) * ( (p*cos(f)-2*r*e)*aR - (p+r)*sin(f)*aT );

param_dot = [dadt;dedt;didt;dodt;dwdt;dmdt];

end
```

Appendix C: Code for Problem 3

```

%% AE 502 HW2 Problem 3, Spring 2021
%   Tiger Hou
close all
clear;clc
latexify

%% Setup
% Earth orbit general parameters
mu = 3.986e14; % m^3/s^2, Earth gravitational parameter
J2 = 1082.63e-6; % J2 perturbation coefficient
req = 6378.137e3; % km, equatorial radius

% initial conditions
a = 7000e3;
e = 0.05;
i = deg2rad(45);
o = deg2rad(0);
w = deg2rad(45);
M0 = deg2rad(0);
E0 = kepler(M0,e);
f0 = 2 * atan(sqrt((1+e)/(1-e))*tan(E0/2));

% find initial position and velocity vectorsr0
[r0,v0] = Get_Orb_Vects([a,e,i,o,w,f0],mu);

% define the J2 perturbation equation as p_J2(r)
p_J2 = @(r) -3/2 * J2 * (mu/norm(r)^2) * (req/norm(r))^2 * ...
    [ ( 1-5*(r(3)/norm(r))^2 ) * r(1)/norm(r); ...
      ( 1-5*(r(3)/norm(r))^2 ) * r(2)/norm(r); ...
      ( 3-5*(r(3)/norm(r))^2 ) * r(3)/norm(r) ];

% Vallado exponential drag model (taking only relevant portions)
drag_Vallado = [200e3,  250e3, 2.789e-10, 37.105e3; ...
                250e3,  300e3, 7.248e-11, 45.546e3; ...
                300e3,  350e3, 2.418e-11, 53.628e3; ...
                350e3,  400e3, 9.518e-12, 53.298e3; ...
                400e3,  450e3, 3.725e-12, 58.515e3; ...
```

```
                 450e3,  500e3, 1.585e-12, 60.828e3; ...
                 500e3,  600e3, 6.967e-13, 63.822e3; ...
                 600e3,  700e3, 1.454e-13, 71.835e3; ...
                 700e3,  800e3, 3.614e-14, 88.667e3; ...
                 800e3,  900e3, 1.170e-14, 124.64e3; ...
                 900e3, 1000e3, 5.245e-15, 181.05e3];
rho = @(r) sum(...
           ( (r-req) >= drag_Vallado(:,1) ) ...
        .* ( (r-req) <  drag_Vallado(:,2) ) ...
        .* drag_Vallado(:,3) ...
        .* exp(-(r-req-drag_Vallado(:,1)) ./ drag_Vallado(:,4)) ...
               ) ...
        /  sum(... this line checks if at least one drag model is matched
           ( (r-req) >= drag_Vallado(:,1) ) ... otherwise division by zero
        .* ( (r-req) <  drag_Vallado(:,2) ) );
% drag model
Cd = 2.0;
A = 5; % m^2
m = 600; % kg
% define the drag perturbation equation as p_drag(rv)
p_drag = @(rv) -1/2 * Cd * A / m * rho(norm(rv(1:3))) ...
            * norm(rv(4:6)) * rv(4:6);

% define overall perturbation model
p = @(rv) p_J2(rv(1:3)) + p_drag(rv);

% calculate orbit period
T = 2*pi * sqrt(a^3/mu); % seconds

%% propagate for J2 + drag case
% ode45
rv0 = [r0;v0];
options = odeset('RelTol',1e-9,'AbsTol',1e-12);
[tDrag,rvDrag] = ode45(@(t,rv)ff(rv,mu,p),[0,10*T],rv0,options);

% plotting
rvDrag = rvDrag';
plot3(rvDrag(1,:), rvDrag(2,:),rvDrag(3,:))
axis equal
```

```

% convert position, velcoity data into orbit parameters
N = size(rvDrag,2);
paramDrag = nan(size(rvDrag));
for j = 1:N
    [a_,e_,i_,o_,w_,f_] = Get_Orb_Params(rvDrag(1:3,j),rvDrag(4:6,j),mu);
    E_ = 2 * atan(sqrt((1-norm(e_))/(1+norm(e_)))*tan(f_/2));
    M_ = E_ - norm(e_)*sin(E_);
    paramDrag(:,j) = [a_,norm(e_),i_,o_,w_,M_]';
end

%% propagate for J2 (without drag)
% ode45
rv0 = [r0;v0];
options = odeset('RelTol',1e-9,'AbsTol',1e-12);
[tNoDrag,rvNoDrag] = ode45(@(t,rv)ff(rv,mu,p_J2),tDrag,rv0,options);

% plotting
rvNoDrag = rvNoDrag';
plot3(rvNoDrag(1,:), rvNoDrag(2,:),rvNoDrag(3,:))
axis equal

% convert position, velcoity data into orbit parameters
N = size(rvNoDrag,2);
paramNoDrag = nan(size(rvNoDrag));
for j = 1:N
    [a_,e_,i_,o_,w_,f_] = Get_Orb_Params(rvNoDrag(1:3,j),rvNoDrag(4:6,j),
        mu);
    E_ = 2 * atan(sqrt((1-norm(e_))/(1+norm(e_)))*tan(f_/2));
    M_ = E_ - norm(e_)*sin(E_);
    paramNoDrag(:,j) = [a_,norm(e_),i_,o_,w_,M_]';
end

%% Compare results
ylabel_vec = {'Semi-Major Axis, m', ...
              'Eccentricity', ...
              'Inclination, deg', ...
              'RAAN, deg', ...
              'Argument of Periapsis, deg', ...
              'Mean Anomaly, deg'};
```

```
xlabel_val = 'Time, s';
for j = 1:6
    subplot(3,2,j)
    delta = paramDrag(j,:)-paramNoDrag(j,:);
    if j == 6 % correction for mean anomaly loopback from 2*pi to 0
        delta = mod(delta,2*pi);
    end
    if j >= 3 % conversion to degrees
        delta = rad2deg(delta);
    end
    plot(tDrag,delta,'LineWidth',1.0)
    xlabel(xlabel_val)
    ylabel(ylabel_vec{j})
    setgrid
end
latexify(20,20)

%% function definitions
function rv_dot = ff(rv,mu,p)
% takes the 6x1 position & velocity vector and computes the derivative
%   also applies perturbation in the form of a function handle p
%   which takes argument p(rv)

rv_dot = nan(6,1);

% velocity
rv_dot(1:3) = rv(4:6);
% acceleration
rv_dot(4:6) = -mu/norm(rv(1:3))^3*rv(1:3) + p(rv);

end
```