ASE 389P.4 Methods of Orbit Determination
Homework 2: Orbit Propagation with Perturbations

Junette Hsin
Masters Student, Aerospace Engineering and Engineering Mechanics, University of Texas, Austin, TX 78712

New force models have been added to the orbit propagator created in Homework 1. The effects that these new forces have on the Keplerian orbit elements and the total specific energy were analyzed.

Introduction

Problem 1

Integrate the equations of motion for one day using the same initial conditions as in Homework #1. However, now include the Earth's oblateness, i.e., the J_2 term. The equations of motion are still $\ddot{\underline{r}} = \nabla U$, but now U includes J_2:

$$U = U_{\text{point mass}} + U_{J_2} = \frac{\mu}{r}\left[1 - J_2\left(\frac{R_{\text{Earth}}}{r}\right)^2\left(\frac{3}{2}\sin^2(\phi) - \frac{1}{2}\right)\right] \tag{1}$$

Use the following (note that some of these values are different from Homework #1!):

$$\begin{aligned} J_2 &= 0.00108248 \\ \mu &= 398600.4 \text{ km}^3/\text{s}^2 \\ R_{\text{Earth}} &= 6378.145 \text{ km} \\ \phi &= \text{Latitude} \rightarrow \sin(\phi) = \frac{z}{r} \\ r &= \sqrt{x^2 + y^2 + z^2} \end{aligned}$$

Use the initial conditions from Homework #1, which are:

$$\underline{r} = -2436.45\hat{i} - 2436.45\hat{j} + 6891.037\hat{k}\ km \tag{2}$$

$$\underline{v} = \dot{\underline{r}} = 5.088611\hat{i} - 5.088611\hat{j} + 0.0\hat{k}\ km/s \tag{3}$$

The state at the final time is posted below to verify that your integrator is working properly.

Solution

Problem 1a

Use the MATLAB symbolic toolbox to compute the Cartesian partial derivatives of U. Compute $\partial U/\partial x$ by hand (include the derivation in the write-up) and compare your results with MATLAB. The full derivation may be scanned and turned in as an appendix, but include the final equation in the typed write-up.

Solution

Problem 1b

Plot the orbital elements a, e, i, Ω, ω, and T_p for one day at 20 second intervals, where T_p is the time of perigee passage and include the figure in your write-up. Be sure to label your axes and ensure that everything in each figure is readable. Using your insight from the two-body model, what conclusions can you draw about the J_2 effect on Keplerian orbital elements? Compute the period of the orbit ($P = 2\pi\sqrt{a^3/\mu}$) based on the initial state (Eqs (2)-(3)). How does the trend in the plots compare to the period?

Solution

Problem 1c

Compute the specific energy (energy/mass) and show that it is conserved around the orbit by plotting $dE = E(t) - E(t_0)$. Use the following equation with U from above:

$$E = \frac{v^2}{2} - U \tag{4}$$

Solution

Problem 1d

Compute h_k, the k-component of the angular momentum vector

$$\underline{h} = \underline{r} \times \underline{v} \tag{5}$$

and plot $dh_k = h_k(t) - h_k(t_0)$ to show that it remains constant.

Solution

Problem 2

Integrate the equations of motion with the conditions given in Problem 1 that include the Earth point-mass, J_2, and now also drag. Use the following relationship for the acceleration due to drag:

$$\ddot{\underline{r}} = -\frac{1}{2} C_D \frac{A}{m} \rho_A V_A \underline{V}_A \tag{6}$$

where

$$\begin{aligned}
C_D &= 2.0 \\
A &= 3.6 \text{ m}^2 \\
m &= 1350 \text{ kg} \\
\rho_0 &= 4 \times 10^{-13} \text{ kg/m}^3 \\
r_0 &= 7298.145 \text{ km} \\
H &= 200.0 \text{ km} \\
\dot{\theta} &= 7.29211585530066 \times 10^{-5} \text{ rad/s} \\
\rho_A &= \rho_0 e^{-(r-r_0)/H} \\
\underline{V}_A &= \begin{bmatrix} \dot{x} + \dot{\theta} y \\ \dot{y} - \dot{\theta} x \\ \dot{z} \end{bmatrix} \\
V_A &= \sqrt{\left(\dot{x} + \dot{\theta} y\right)^2 + \left(\dot{y} - \dot{\theta} x\right)^2 + \dot{z}^2}
\end{aligned}$$

Make sure you have agreement in all of your units when computing the drag acceleration! Compare your result to the online values to ensure that your integrator is working properly.

Solution

Problem 2a

Compute the specific energy at 20 second intervals and plot $dE = E(t) - E(t_0)$ (using Eq. (4)). Include the plot in your write-up. What can you infer from the plot? Is the total energy conserved? Why or why not?

Solution

Problem 2b

Compute the same Keplerian orbital elements generated in Problem 1b at 20 second intervals (a, e, i, Ω, ω, and T_p) and plot the differences in these elements from those computed in Problem 1. That is, generate time histories of each orbital element with and without drag and plot the differences, e.g., $a_{2B+J2+Drag} - a_{2B+J2}$. Include the results in your write-up. What can you observe in these plots? Which orbital elements are impacted by drag and how are they affected?

Solution

Conclusion

Appendix

HW1 MATLAB code

```
% ASE 389 Orbit Determination
% HW 1
% Junette Hsin

%% Problem 1

global mu

mu = 398600.5 ;
r  = [ -2436.45; -2436.45; 6891.037 ];
v  = [ 5.088611; 5.088611; 0 ];

rv = [r; v];
oe = rv2oe(rv);

%% Problem 2

rv = oe2rv(oe);

%% Problem 3

x = -2436.45;
y = -2436.45;
z = 6891.037;
mu = 398600.5;

dux = -2*mu*x / ( x^2 + y^2 + z^2 )^2;
duy = -2*mu*y / ( x^2 + y^2 + z^2 )^2;
duz = -2*mu*z / ( x^2 + y^2 + z^2 )^2;

rnorm = sqrt( x^2 + y^2 + z^2 );

dux = -mu*x / ( rnorm )^3;
duy = -mu*y / ( rnorm )^3;
duz = -mu*z / ( rnorm )^3;

%% Problem 4

a = oe(1);
T   = abs(2 * pi * sqrt(a^3 / mu));        % period

toler   = 1e-8;          % 1e-14 accurate; 1e-6 coarse
options = odeset('reltol', toler, 'abstol', toler );
[t,x] = ode45(@TwoBod_6states, [0 2*T], [r; v], options);

for i = 1:length(t)
rnorm(i) = norm(x(i, 1:3));
vnorm(i) = norm(x(i, 4:6));
H(i, :) = cross(x(i, 1:3), x(i, 4:6));
hnorm(i) = norm(H(i, :));
end

anorm = 0;
for i = 2:length(t)
a = (x(i, 4:6) - x(i-1, 4:6)) / ( t(i) - t(i-1) );
anorm(i) = norm(a);
end

% ---------------------------------------------------------------

name = 'Problem 4: 2-Body EOM';
h = figure('name', name);

% position
```

```matlab
subplot(3,1,1)
plot(t, rnorm); grid on
title('r norm')
ylabel('km')

% velocity
subplot(3,1,2)
plot(t, vnorm); grid on
title('v norm')
ylabel('km/s')

% acceleration
subplot(3,1,3)
plot(t, anorm); grid on
title('a norm');
ylabel('km/s^2')
xlabel('time (sec)')

sgtitle(name)

save_pdf(h, 'prob4_2bodeom');

% --------------------------------------------------------------------------

name = 'Problem 4: 2-Body EOM Orbit';
h = figure('name', name);
plot3(x(:,1), x(:,2), x(:,3)); hold on; grid on;
plot3(x(1,1), x(1,2), x(1,3), 'o')
plot3(x(end,1), x(end,2), x(end,3), 'x')
xlabel('x (km)')
ylabel('y (km)')
zlabel('z (km)')
legend('orbit', 'start', 'end')

sgtitle(name)

save_pdf(h, 'prob4_2bodeom_orbit');

% --------------------------------------------------------------------------

clear oe
for i = 1:length(t)
oe(i,:) = rv2oe(x(i,:));
end

labels = {'a', 'e', 'i', '\omega', '\Omega', '\nu'};
units = {'km', '', 'rad', 'rad', 'rad', 'rad'};
name = 'Problem 4: 2-Body Orbital Elements';
h = figure('name', name, 'position', [100 100 500 600]);
for i = 1:6
subplot(6,1,i)
plot(t, oe(:, i)); grid on
title(labels{i});
ylabel(units{i});
end
xlabel('time (sec)')
sgtitle(name)

save_pdf(h, 'prob4_2bodoes');

% --------------------------------------------------------------------------

name = 'Problem 4: 2-Body Specific Angular Momentum';
h = figure('name', name);
subplot(2,1,1)
scatter3(H(:,1), H(:,2), H(:,3)); grid on
xlabel('x (km^2/s)')
ylabel('y (km^2/s)')
```

```
zlabel('z (km^2/s)')
title('h (scatter plot)')

subplot(2,1,2)
plot(t, hnorm); grid on
xlabel('time (sec)')
ylabel('km^2/s')
title('h norm vs time')

sgtitle(name)

save_pdf(h, 'prob4_angmom')

%% Problem 5

% specific kinetic energy
for i = 1:length(t)
T(i) = 0.5 * vnorm(i)^2;
U(i) = mu / rnorm(i);
end
E = T - U;

name = 'Problem 4: 2-Body Specific Energy';
h = figure('name', name);
subplot(2,1,1)
plot(t, E); grid on; hold on;
plot(t, T);
plot(t, U);
ylabel('km^2/s^2')
legend('Total', 'Kinetic', 'Potential')
title('Total Specific Energy: Kinetic - Potential')
subplot(2,1,2)
plot(t, [0 diff(E)]); grid on
title('Change in Total Specific Energy')
xlabel('Time (sec)')
ylabel('km^2/s^2')
sgtitle(name)

save_pdf(h, 'prob5_energy')

%% subfunctions

function save_pdf(h, name)

% save as cropped pdf
set(h,'Units','Inches');
pos = get(h,'Position');
set(h,'PaperPositionMode','Auto','PaperUnits','Inches','PaperSize',[pos(3), pos(4)])
print(h,name,'-dpdf','-r0')

end
```

rv2oe function

```
function oe = rv2oe(rv)
% ---------------------------------------------------------------------
% Inputs
%   rv = [6x1] position and velocity states vector
%
% Outputs
%   oe = [6x1] orbital elements: a, e, i, w, Omega, nu
%           a       = semimajor axis
%           e       = eccentricity
%           i       = inclination
%           w       = argument of perigee
%           Omega   = right ascension of ascending node
%           nu      = true anomaly
```

```
% --------------------------------------------------------------------------

global mu

r = rv(1:3);
v = rv(4:6);

% angular momentum
h       = cross(r,v);

% node vector
nhat    = cross( [0 0 1], h );

% eccentricity
evec    = ( (norm(v)^2 - mu/norm(r))*r - dot(r,v)*v ) / mu;
e       = norm(evec);

% specific mechanical energy
energy  = norm(v)^2/2 - mu/norm(r);

% semi-major axis and p
if abs(e-1.0)>eps
a = -mu/(2*energy);
p = a*(1-e^2);
else
p = norm(h)^2/mu;
a = inf;
end

% inclination
i = acos(h(3)/norm(h));

% right ascension of ascending node (check for equatorial orbit)
if i > 0.000001
Omega = acos( nhat(1)/norm(nhat) );
else
Omega = 0;
end
if isnan(Omega)
Omega = 0;
end
if nhat(2)<0
Omega = 2*pi - Omega;
end

% argument of perigee
if e > 0.000001
w = acos(dot(nhat,evec)/(norm(nhat)*e));
else
w = 0;
end
if isnan(w)
w = 0;
end
% if e(3)<0
%    argp = 360-argp
% end

% true anomaly
nu = acos( dot(evec,r) / (e*norm(r)) );
% if dot(r,v)<0
%    nu = 360 - nu
% end

oe = [a; e; i; w; Omega; nu];

end
```

oe2rv function

```
function [rv] = oe2rv(oe)
% ------------------------------------------------------------------------
% Purpose: Convert orbital elements and time past epoch to the classic
% Cartesian position and velocity
%
% Inputs:
%    oe       = [6x1] or [1x6] orbital elements
%    delta_t = t - t0 time interval
%    mu       = Gravity * Mass (of Earth) constant
%
% Outputs:
%    rv       = position and velocity state vector
% ------------------------------------------------------------------------

% global delta_t
global mu

a        = oe(1);
e        = oe(2);
i        = oe(3);
w        = oe(4);
LAN      = oe(5);
% M0       = oe(6);
nu       = oe(6);

% nu is TRUE ANOMALY --> use Kepler's to calculate MEAN ANOMALY
% E = 2*atan( sqrt( (1-e)/(1+e) ) * tan(nu/2) );
% M = M0 + sqrt( mu/a^3 ) * (delta_t);
% E = keplerEq(M, e, eps);
% E = kepler(M, e);
% nu = 2*atan( sqrt( (1+e)/(1-e) ) * tan(E/2) );

p = a * ( 1 - e^2 );              % intermediate variable
r = p / ( 1 + e*cos(nu) );        % r_magnitude, polar coordinates

% Perifocal position and velocity

r_pf = [ r * cos(nu); r * sin(nu); 0 ];
v_pf = [ -sqrt(mu/p) * sin(nu); sqrt(mu/p) * (e + cos(nu)); 0 ];

% Perifocal to ECI transformation, 3-1-3 rotation
R11 = cos(LAN)*cos(w) - sin(LAN)*sin(w)*cos(i);
R12 = -cos(LAN)*sin(w) - sin(LAN)*cos(w)*cos(i);
R13 = sin(LAN)*sin(i);

R21 = sin(LAN)*cos(w) + cos(LAN)*sin(w)*cos(i);
R22 = -sin(LAN)*sin(w) + cos(LAN)*cos(w)*cos(i);
R23 = -cos(LAN)*sin(i);

R31 = sin(w)*sin(i);
R32 = cos(w)*sin(i);
R33 = cos(i);

R = [R11 R12 R13; R21 R22 R23; R31 R32 R33];

% Transform perifocal to ECI frame
r_vec = R * r_pf;
v_vec = R * v_pf;

% Position and state vector
rv = [r_vec; v_vec];

end

%% Kepler equation solvers

```

```
function E = keplerEq(M,e,eps)
% Function solves Kepler's equation M = E-e*sin(E)
% Input - Mean anomaly M [rad] , Eccentricity e and Epsilon
% Output  eccentric anomaly E [rad].
En  = M;
Ens = En - (En-e*sin(En)- M)/(1 - e*cos(En));
while ( abs(Ens-En) > eps )
En = Ens;
Ens = En - (En - e*sin(En) - M)/(1 - e*cos(En));
end
E = Ens;
end

function E = kepler(M, e)
f = @(E) E - e * sin(E) - M;
E = fzero(f, M);  % <-- I would use M as the initial guess instead of 0
end
```

s

References

[1] Donald D. Mueller, J. W., and Bate, R. R., *Fundamentals of Astrodynamics*, Dover Publications, Inc., 1971.

[2] Jah, M. K., "ASE 389P.4 Methods of Orbit Determination Module 3," , January 2021.

[3] Jah, M. K., "ASE 389P.4 Methods of Orbit Determination Module 2," , January 2021.