AAE 440: Spacecraft Attitude Dynamics

PS10

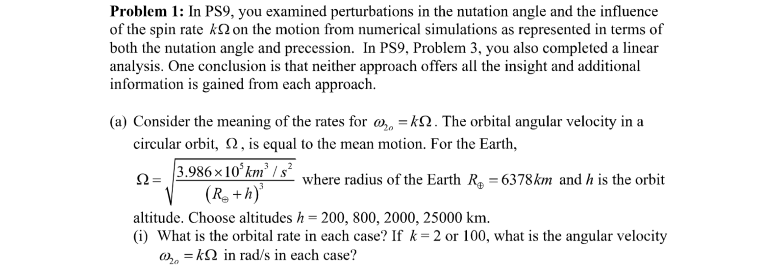
Dr. Howell

School of Aeronautical and Astronautical

Purdue University

*Tomoki Koike*

*Monday April 20th, 2020*

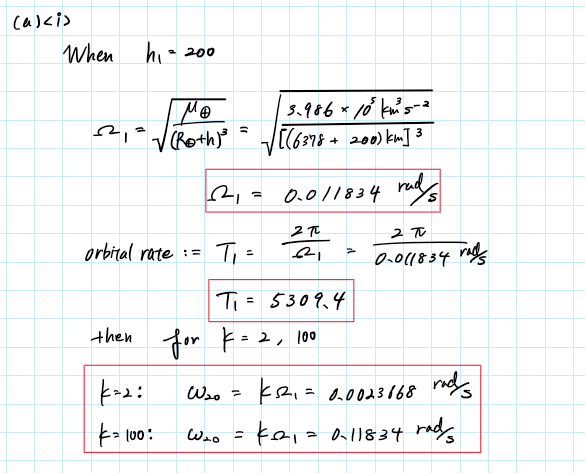


By plugging in the altitude values h = [200, 800, 2000, 25000] into the equation

And then, using the output we can compute the orbital rate from the following equation

Next, we take the product of the spin factor k and to find the angular velocity.

The calculations for when h = 200 km becomes,



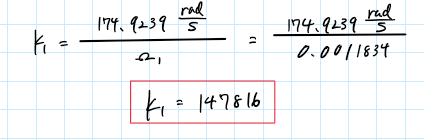
With the same procedure we can obtain the rest. Then, we get the following tabulated results.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| [km] | [rad/s] | Orbital Rate [s] |  |  |
| 200 | 0.0011834 | 5309.4 | 0.0023668 | 0.11834 |
| 800 | 0.0010382 | 6052.2 | 0.0020763 | 0.10382 |
| 2000 | 0.00082330 | 7631.7 | 0.0016466 | 0.082330 |
| 25000 | 0.00011359 | 55315.8 | 0.00022718 | 0.011359 |



If

Example, calculations for the first altitude values becomes,



Thus, by plugging in the numbers we get the values for spin factor kj

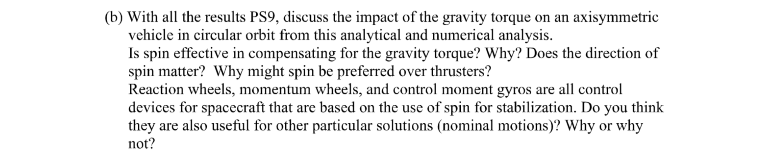
|  |  |
| --- | --- |
| [rad/s] |  |
| 0.0011834 | 41.0600 |
| 0.0010382 | 46.8040 |
| 0.00082330 | 59.0185 |
| 0.00011359 | 427.7754 |



One example, of a spin stabilized spacecraft is the Lunar Prospector.

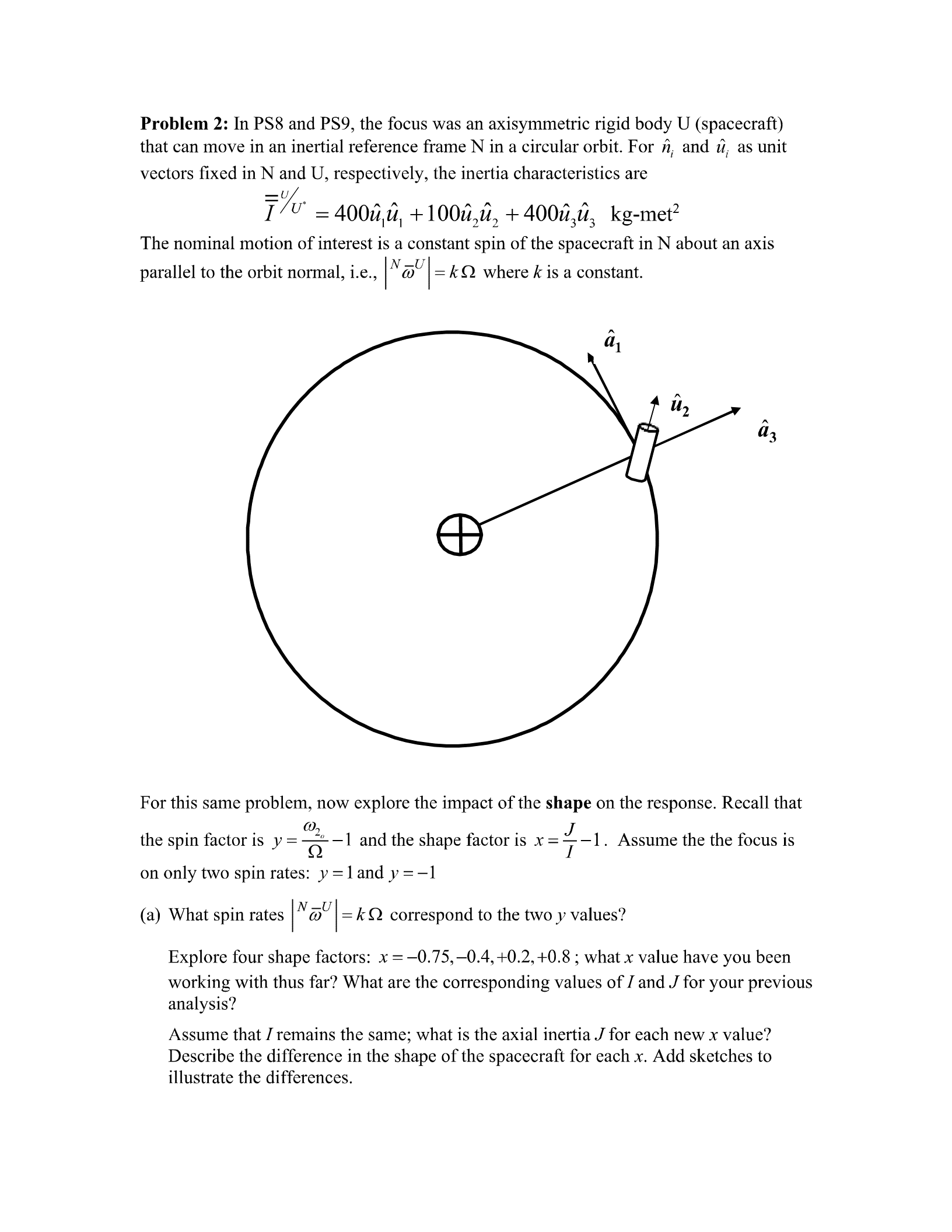
On NASA’s archive it is noted that the nominal spin rate of the Lunar Prospector is 12 rpm.

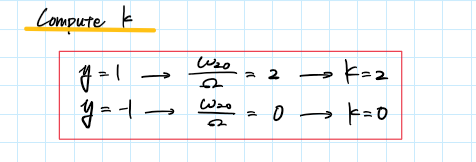
(NASA).



Discussion

* Spin enables the spacecraft to resist perturbations such as gravity force, and therefore, is effective.
* The direction of the spin matters depending on the magnitude of the spin rate. With a higher spin rate, the direction does not matter as much, whereas for low spin rates the perturbation acts more strongly on the spacecraft and the effect can be mitigated as well as exacerbated based on the direction of rotation. This has been observed from our analysis in PS9\*.
* Spin stabilization plays a pivotal role in satellites that are expected to remain in a stable inertially fixed direction. Reaction wheels, momentum wheels, gyros, etc. enable for example weather satellites or surveillance satellites to tidally lock itself and improve data acquisition. Other methods such as thrusters are implemented as well to cope with perturbation.
* As we have analyzed in PS9\* problem 3 by analytically assessing the stability by linearizing the Euler equation based on a nominal motion. From that we were able to correlate the stability and the spin factor clearly, which shows that spin stabilization and control methods are very important for other particular solutions.





Compute J

Using the relation

We can figure out the x, I, J, and the shape of the spacecraft. The tabulated results are the following.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x | I [ kg-m2] | J [kg-m2] | Shape of Spacecraft | |
| -0.75 | 400 | 100 | I > J | Rod-like |
| -0.4 | 400 | 240 | I > J | Rod-like |
| 0.2 | 400 | 480 | J > I | Disk-like |
| 0.8 | 400 | 720 | J > I | Disk-like |

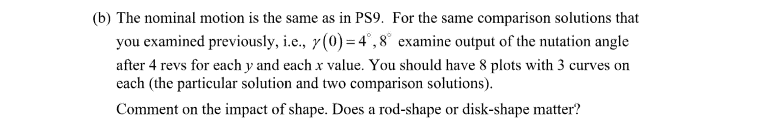
Illustrations

x = -0.4

x = -0.75

x = 0.2

x = 0.8



Simulation Plots

A close up of a map

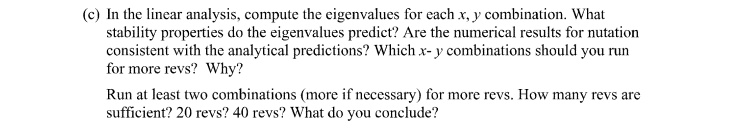
Description automatically generated

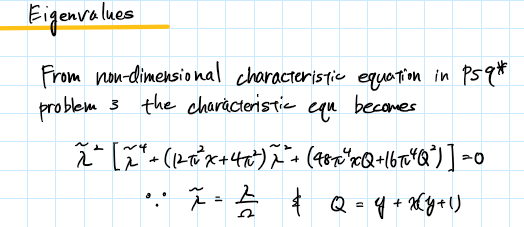
A close up of a map

Description automatically generated

Analysis

* Observing the case of k = 0 and y = -1, we can see that it shows instabilities for any x-value besides the value of x =0.2 This shows that the shape of the body does not effect the motion, but for only for small positive values of x.
* Next, observing the case of k=2 and y=1, only for these specific can we tell that the shape of the motion matters for the stability. For example, when the shape is disk-like the motion is stable, whereas when the shape is rod-like the motion is unstable. Additionally, the plot tells us that as the period of the patterns increase the x-value decreases.





Now, solving using MATLAB (code in appendix) by plugging in the provided values, we get the tabulated results on the next page.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **y** | **x** |  | **Prediction** | **Numerical Results** | **Linear Analytical Analysis** | **Congruency** |
| 1 | -0.75 | +2.9426i | Inconclusive | Probably Unstable | Marginally Stable | Yes |
| -2.9426i |
| +1.2253i |
| -1.2253i |
| -0.4 | 0 | Inconclusive | Probably Unstable | Marginally Stable | Yes |
| 0 |
| 0 |
| 0 |
| 0.2 | 0 | Inconclusive | Probably Stable | Marginally Stable | Yes |
| 0 |
| 0 |
| 0 |
| 0.8 | 0 | Inconclusive | Probably Stable | Marginally Stable | Yes |
| 0 |
| 0 |
| 0 |
| -1 | -0.75 | +2.9426i | Inconclusive | Probably Unstable | Marginally Stable | Yes |
| -2.9426i |
| +1.2253i |
| -1.2253i |
| -0.4 | +0.54589 | Unstable | Probably Unstable | Unstable | Yes |
| -0.54589 |
| +2.1675i |
| -2.1675i |
| 0.2 | 0 | Inconclusive | Probably Stable | Marginally Stable | Yes |
| 0 |
| 0 |
| 0 |
| 0.8 | 0 | Inconclusive | Probably Unstable | Marginally Stable | Yes |
| 0 |
| 0 |
| 0 |

Tabulated Results

Simulation Plots

A screenshot of a cell phone

Description automatically generated

A screenshot of a social media post

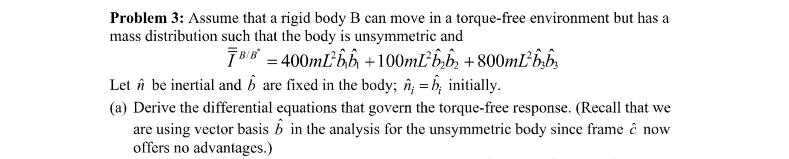
Description automatically generated

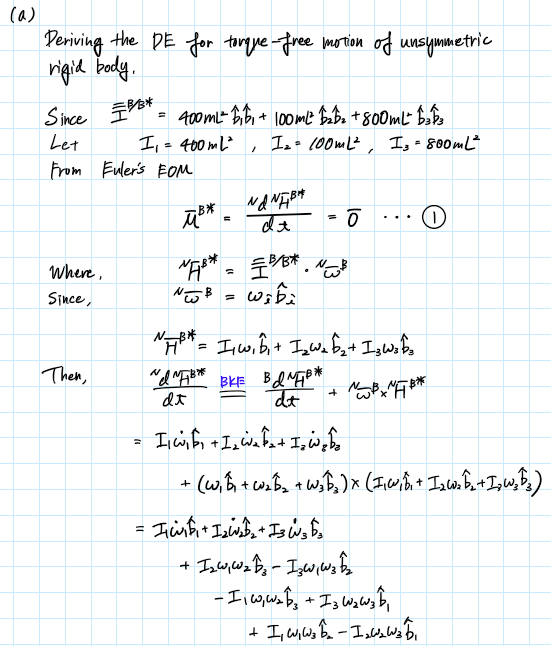
A screenshot of a social media post

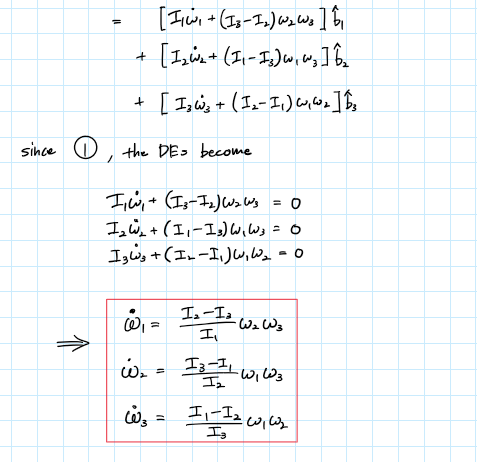
Description automatically generated

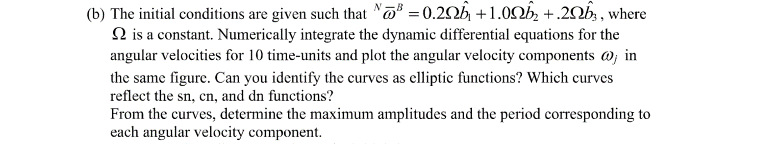
Analysis

* Every plot shows no large change in the pattern of the nutation angles for both 20 revolutions and 40 revolutions. The angles for both simulated revolutions, the nutation angle amplitudes seem to remain within the range of which can be identified as stable.
* Though, we cannot conclude the system to be stable for the lack of information; however, we can say that the system is possibly/potentially stable.









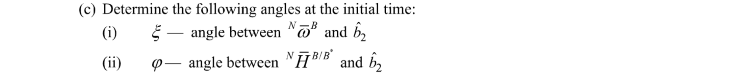
A close up of a map

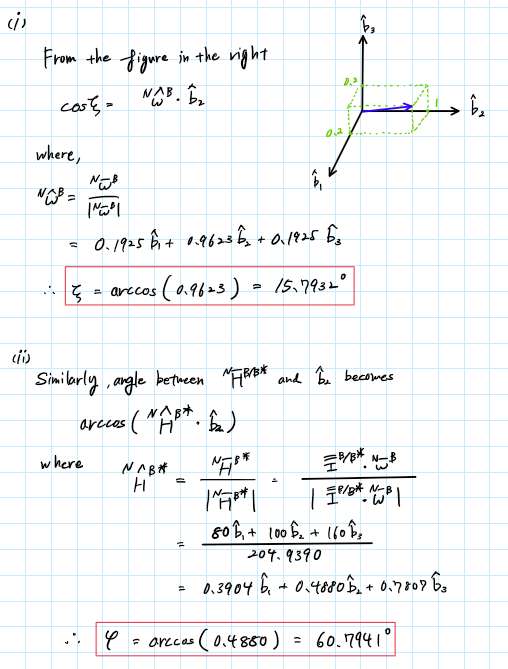
Description automatically generated

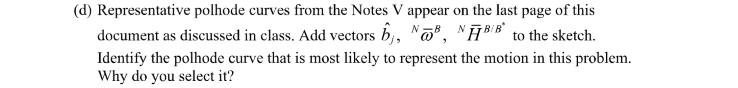
Analysis

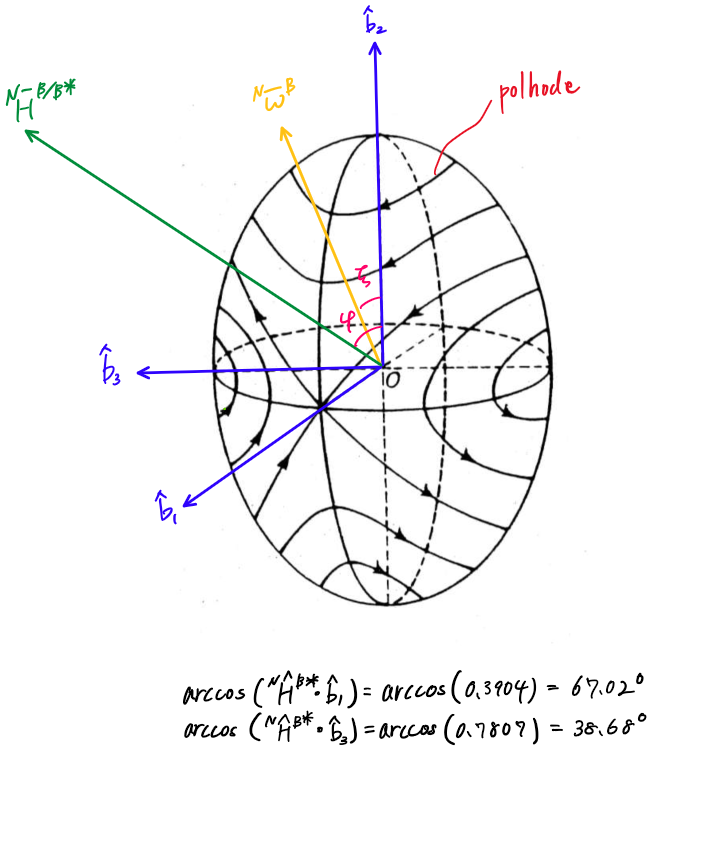
* From this plot, we can identify the angular velocities as a Jacobi elliptic functions because the plots are periodic and are evidently sinusoidal.
* By drawing a vertical line at the point (0.5, 0), we can see that decreases right after the initial point. This means that is the elliptical function. Then because increases right after the initial point, this means that is sn(x, k). Thus, the remaining which is somewhat distant from the other two becomes dn(x, k).
* Using MATLAB (code in Appendix) we can calculate the maximum amplitude by taking the difference of the maximum value and minimum value of each angular velocity waves and dividing that by 2. The results are tabulated below.

|  |  |
| --- | --- |
|  | Maximum Amplitude [rad/s] |
|  | 0.4761 |
|  | 0.1438 |
|  | 0.2204 |









Discussion

* The specific polhode was selected because it is the only one that intersects with the angular velocity vector that we have sketched on the ellipsoid diagram. We also know that the movement is confined to one curve because it depends on the initial conditions of the angular velocity and since the largest angular velocity is the component, and therefore the encirclement will tend to be near and around the -axis.

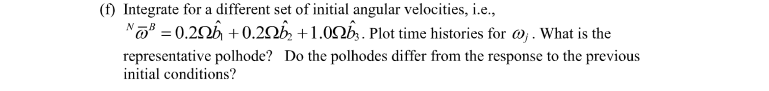


A close up of text on a white background

Description automatically generated

Analysis

* This system is marginally stable by our definition because the rotational axis is near the minimum moment of inertia which allows the rotational axis to remain close to the principle direction with the minimum moment of inertia. The closer the angular velocity vector is to the min/max moment of inertia the smaller the deviation in angle for each cycle.



Plot

A close up of text on a white background

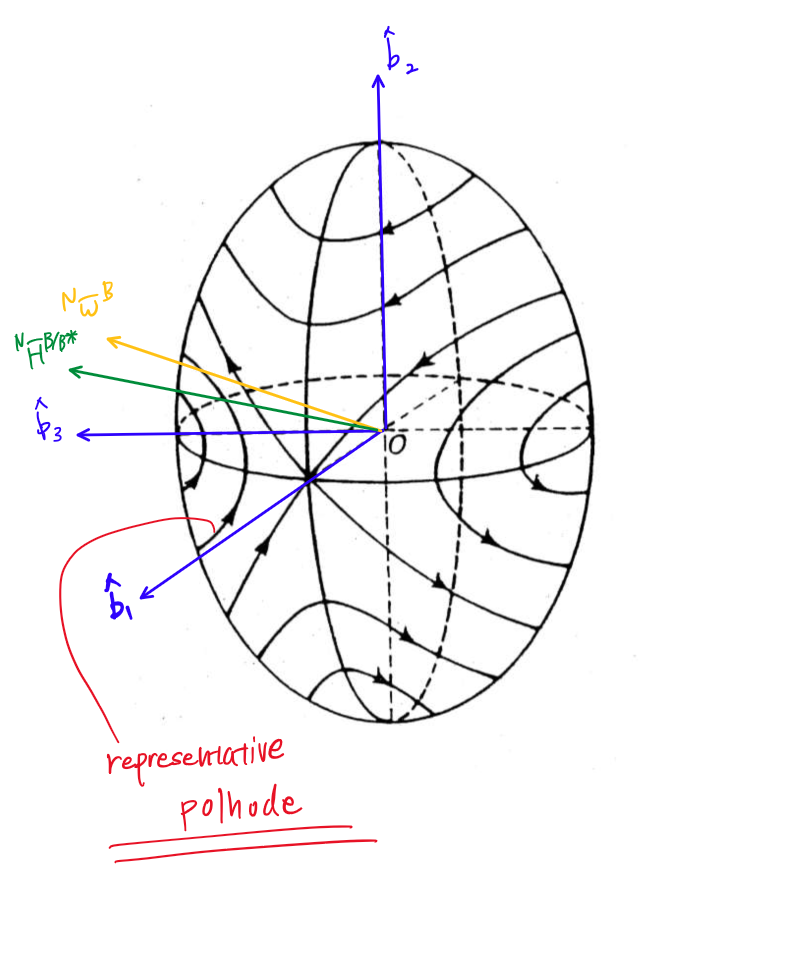
Description automatically generated

With the same procedure in problem (d) we find out that the angles between and each axis , , and as well as and each axis , , and are the following,

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | |  | | |
| , | , |  | , | , |  |
| 78.9402o | 78.9402o | 15.7932o | 84.2912o | 88.5750o | 5.8851o |

Based on these angles we can sketch and onto the ellipsoid like how we did in problem (d).

Representative Polhode



Analysis

* The polhodes differ from what we had for the first initial conditions. This is because for the second initial conditions the angular velocity was the closest to -axis due to its largest component being the component. Thus, the rotation will tend to trace a path near the -axis.

Appendix

## **AAE 440 PS10 Problem 1**

clear all; close all; clc;

fdir = 'C:\Users\Tomo\Desktop\studies\2020-Spring\AAE440\MATLAB\outputs\HW10';

set(groot, 'defaulttextinterpreter','latex');

set(groot, 'defaultAxesTickLabelInterpreter','latex');

set(groot, 'defaultLegendInterpreter','latex');

### (a)

#### (i)

% Given properties

Re = 6378; % radius of the Earth [km]

mu\_e = 3.986e5; % [m3s-2]

h = [200 800 2000 25000]; % altitudes [km]

k = [2; 100]; % spin factor

% Anonymous function for mean motion Omega

Omega = @(h) sqrt(mu\_e./(Re + h).^3);

% Orbital rates

OMG\_i = Omega(h)

T\_i = 2\*pi./OMG\_i

% Angular velocities

w\_2o = k.\*OMG\_i

#### (ii)

w\_2o = 0.464\*2\*pi/60;

k\_j = w\_2o./OMG\_i

## **AAE 440 PS10 Problem 2-2**

clear all; close all; clc;

fdir = 'C:\Users\Tomo\Desktop\studies\2020-Spring\AAE440\MATLAB\outputs\HW10';

set(groot, 'defaulttextinterpreter','latex');

set(groot, 'defaultAxesTickLabelInterpreter','latex');

set(groot, 'defaultLegendInterpreter','latex');

% Defining System Properties

I = 400; % transverse moment of inertia [kg m2]

rev = 0:0.001:4; % Integration revolutions

x = [-0.75 -0.4 0.2 0.8]; % shape factor

% x = [-0.01 0.01 0.1 0.2];

y = [1 -1];

## **(a)**

% spin factor

k = y + 1;

% axial moment of inertia [kg m2]

J = I.\*(1+x);

## **(b)**

% Initial perturbation [deg]

gamma0 = [0 4 8];

peak = zeros(1,3);

for i = 1:length(k) % spin factors

fig = figure("Renderer","painters","Position",[10 10 900 700]);

fig.NumberTitle = i;

for j = 1:length(J) % shape factors

for n = 1:length(gamma0) % initial perturbations

% Nutation History

[rev, Nut] = Nut\_History(rev,k(i),gamma0(n),I,J(j));

peak(n) = max(Nut);

% Plotting Gammas

subplot(2,2,j);

hold on;

plot(rev, Nut, 'DisplayName',['$\gamma\_0$ = ',num2str(gamma0(n)), '$^o$'])

end

grid on; grid minor; box on; ylim([-5 max(peak)+15]);

legend;

str = sprintf('$k=%d$ and $x=%.2f$',k(i),x(j));

title(str)

end

% Give common xlabel, ylabel and title to figure

han = axes(fig,'Visible','off');

han.XLabel.Visible = 'on';

han.YLabel.Visible = 'on';

han.Title.Visible = 'on';

xlabel(han,'Revolutions');

ylabel(han,'Nutation Angle, $\gamma$ [deg]');

sgt = sgtitle("Nutation Angle Histories, Koike");

sgt.Visible = 'on';

str = sprintf('2b\_Sub\_Nut\_History\_k=%d.png',k(i));

saveas(fig, fullfile(fdir, str));

close all;

end

## **(c)**

% More Revolutions

revN = [20, 40];

gamma0 = [0 4 8];

for i = 1:3

% Combinations to be run for more revolutions

% [y, x] = [1, 0.2], [1, 0.8], [-1, 0.2]

fig = figure("Renderer","painters","Position",[10 10 900 700]);

fig.NumberTitle = i;

switch i

case 1

y = 1; x = 0.2;

case 2

y = 1; x = 0.8;

case 3

y = -1; x = 0.2;

end

k = y + 1; % spin factor

J = I\*(1+x); % axial moment of inertia [kg m2]

for j = 1:2

rev = 0:0.001:revN(j);

for n = 1:length(gamma0) % initial perturbations

% Nutation History

[rev, Nut] = Nut\_History(rev,k,gamma0(n),I,J);

peak(n) = max(Nut); % max nutation

subplot(2,1,j);

hold on;

plot(rev, Nut, 'DisplayName', ...

['$\gamma\_0$ = ',num2str(gamma0(n)), '$^o$'])

end

grid on; grid minor; box on; legend; ylim([-5 max(peak)+5])

str = sprintf('$y=%d$ and $x=%.1f$ for %d rev',y,x,rev(end));

title(str)

end

% Give common xlabel, ylabel and title to figure

han = axes(fig,'visible','off');

han.XLabel.Visible = 'on';

han.YLabel.Visible = 'on';

han.Title.Visible = 'on';

xlabel(han,'Revolutions');

ylabel(han,'Nutation Angle, $\gamma$ [deg]');

sgt = sgtitle("Nutation Angle Histories, Koike");

sgt.Visible = 'on';

str = sprintf('2.(c)Sub\_Nut\_History\_y=%d\_x=%.1f.png',y,x);

saveas(fig, fullfile(fdir, str));

close all;

end

## **AAE 440 PS10 Problem 3**

clear all; close all; clc;

fdir = 'C:\Users\Tomo\Desktop\studies\2020-Spring\AAE440\MATLAB\outputs\HW10';

set(groot, 'defaulttextinterpreter','latex');

set(groot, 'defaultAxesTickLabelInterpreter','latex');

set(groot, 'defaultLegendInterpreter','latex');

### (b)

% Defining System Properties

I = [400 100 800]; % moment of inertia [kg-m2]

Omega = 1;

w0 = Omega\*[0.2 1 0.2]; % initial angular velocity [rad/s]

t\_span = 0:0.001:10; % 10 time-units [s]

% Numerical Integration

opt = odeset('RelTol', 1e-13, 'AbsTol', 1e-13);

[t, data] = ode45(@(t,w) Tfree\_unsymm\_EOM(t,w,I),t\_span,w0,opt);

% Assign variables from the output of the numerical integration

w1 = data(:,1);

w2 = data(:,2);

w3 = data(:,3);

% Plot the angular velocities

fig1 = figure("Renderer","painters");

plot(t,w1)

title({'Angular Velocity History for 10 time-units ($\Omega$=1), Koike'})

xlabel('time [s]')

ylabel('angular velocity [rad/s]')

hold on;

plot(t,w2); plot(t,w3); hold off;

grid on; grid minor; box on;

legend('$w\_1$','$w\_2$','$w\_3$',"Location",'southeast')

saveas(fig1,fullfile(fdir,'P3-b\_angVel.png'));

% Find amplitude for each angular velocity

amp\_w1 = (max(w1)-min(w1))/2

amp\_w2 = (max(w2)-min(w2))/2

amp\_w3 = (max(w3)-min(w3))/2

#### (c)

% Angle between omegaNB and b2 hat

b2\_hat = [0 1 0];

w0\_hat = w0/norm(w0)

zeta = acosd(dot(w0\_hat,b2\_hat))

% Angle between H\_NB and b2 hat

H\_NB = I.\*w0

H\_NB\_hat = H\_NB/norm(H\_NB)

psi = acosd(dot(H\_NB\_hat,b2\_hat))

#### (d)

% Plotting w3 as a function w1

fig2 = figure("Renderer","painters");

plot(w1,w3)

title({'$\omega\_3$ as a Function of $\omega\_1$, Koike'})

xlabel('$\omega\_1$ [rad/s]'); ylabel('$\omega\_3$ [rad/s]')

grid on; grid minor; box on; axis equal;

saveas(fig2,fullfile(fdir,'P3-d\_w1\_vs\_w3.png'));

#### (f)

w0 = Omega\*[0.2 0.2 1]; % initial angular velocity [rad/s]

% Numerical Integration

opt = odeset('RelTol', 1e-13, 'AbsTol', 1e-13);

[t, data] = ode45(@(t,w) Tfree\_unsymm\_EOM(t,w,I),t\_span,w0,opt);

% Assign variables from the output of the numerical integration

w1 = data(:,1);

w2 = data(:,2);

w3 = data(:,3);

% Plot the angular velocities

fig3 = figure("Renderer","painters");

plot(t,w1)

title({'Angular Velocity History for 10 time-units', '($\Omega$=1) Problem (f), Koike'})

xlabel('time [s]')

ylabel('angular velocity [rad/s]')

hold on;

plot(t,w2); plot(t,w3); hold off;

grid on; grid minor; box on;

legend('$w\_1$','$w\_2$','$w\_3$',"Location",'best')

saveas(fig3,fullfile(fdir,'P3-e\_angVel.png'));

% Angle between omegaNB and b2 hat

b2\_hat = [0 1 0];

w0\_hat = w0/norm(w0)

zeta = acosd(dot(w0\_hat,b2\_hat))

acosd(dot(w0\_hat,[1 0 0]))

acosd(dot(w0\_hat,[0 0 1]))

% Angle between H\_NB and b2 hat

H\_NB = I.\*w0

H\_NB\_hat = H\_NB/norm(H\_NB)

psi = acosd(dot(H\_NB\_hat,b2\_hat))

acosd(dot(H\_NB\_hat,[1 0 0]))

acosd(dot(H\_NB\_hat,[0 0 1]))

function lambdas = eigenvalues(I,J,k,type)

%{

Function: eigenvalues

Author: Tomoki Koike

Description: Eigenvalues per omega will be computed.

>>Inputs

I: transverse moment of inertia

J: rotational moment of inertia

k: spin factor k

type: 'd' = dimensional

'nd' = non-dimensional

Outputs<<

lambdas: eigenvalues

%}

x = J/I - 1; % shape factor

y = k - 1; % spin factor

Q = y + (1 + y)\*x;

% Dimensional or Non\_dimensional

if type == 'd'

A = 3\*x+Q^2+1;

B = 3\*x\*Q+Q^2;

else % type == 'nd'

A = (3\*x+Q^2+1)\*4\*pi^2;

B = (3\*x\*Q+Q^2)\*16\*pi^4;

end

% Eigenvalues

lambdas = [ 0 ;

0 ;

sqrt((-A + sqrt(A^2-4\*B))/2);

-sqrt((-A + sqrt(A^2-4\*B))/2);

sqrt((-A - sqrt(A^2-4\*B))/2);

-sqrt((-A - sqrt(A^2-4\*B))/2)];

end

function [rev, Nut, deltaK] = Nut\_History(rev,k,gamma0,I,J)

%{

Function: Nut\_History

Author: Tomoki Koike

Description: This function computes the nutation angle history from

the ode45 simulation for a rigid body motion.

>>Inputs

rev: number of revolutions

k: spin factor

gamma0: initial perturbation [deg]

I: transverse moment of inertia

J: axial moment of inertia

Outputs<<

rev: revolutions

Nut: Nutation angle history

deltaK: Euler Constraint Perturbation

%}

% Initial Conditions

w0 = [0 k 0];

e0 = [sind(gamma0/2) 0 0 cosd(gamma0/2)];

y0 = [w0 e0 0];

% Numerical Integration

opt = odeset('RelTol', 1e-13, 'AbsTol', 1e-13);

[rev, data] = ode45(@(v,y) nond\_EOM(v,y,I,J,k),rev,y0,opt);

% Calculating Nutation angle

Nut = acosd(1-2\*data(:,6).^2-2\*data(:,4).^2);

% Euler Constraint

deltaK = data(:,8);

end

function dwdt = Tfree\_unsymm\_EOM(t,w,I)

%{

Function: unsymm\_Tfree\_EOM

Author: Tomoki Koike

Description: The dynamic differential equation of a torque free

motion for a unsymmetric rigid body.

>>Inputs

t: time span to analyze

w: angular velocities

I: moment of inertia

Outputs<<

dwdt: differential w

%}

dwdt = zeros([3 1]);

% Dynamic differential equations

dwdt(1) = (I(2)-I(3))/I(1)\*w(2)\*w(3);

dwdt(2) = (I(3)-I(1))/I(2)\*w(1)\*w(3);

dwdt(3) = (I(1)-I(2))/I(3)\*w(1)\*w(2);

end

References

Williams, David R. “Lunar Prospector.” *NASA Space Science Data Coordinated Archive*, NASA, 1998, nssdc.gsfc.nasa.gov/nmc/spacecraft/display.action?id=1998-001A.