All codes are matlab codes

E 1

The code used to solve the problem is the following

```
%% Inputs
I11 = 20; % kg m²
I22 = 60; % kg m²
I33 = 60; % kg m²

w = deg2rad([15, 5, -10])'; % rad / s

%% Calculations
% (a)
wp = ((I11/I22) - 1) * w(1);
T_p = abs(2*pi/wp)
% (b)
w_23 = sqrt(w(2)^2 + w(3)^2)
```

```
T_p =
    36 s
```

a

```
w_23 =
    0.1951 rad / s
```

b

E 2

The code used to solve the problem is the following

```
%% Inputs
I11 = 300; % kg m²
I22 = 200; % kg m²
I33 = 100; % kg m²
inertia = [I11  I22 I33];

% state: q0 q1 q2 q3 w1 w2 w3
state = [1, 0, 0, 0, 0.2, -0.3, 0.1];

%% Calculations

tspan = [0 60];
options = odeset('RelTol', 1e-8, 'AbsTol', 1e-8);
[t, c_state] = ode45(@(t, s) quaternion_derivative(s, inertia), tspan, state, options);

err = abs(c_state(:, 1).^2 + c_state(:, 2).^2 + c_state(:, 3).^2 + c_state(:, 4).^2 - 1);

%% Functions
```

```
function dqdt = quaternion_derivative(state, inertia)
    % Split state on its components
    q = state(1:4);
    omega = state(5:7);
    q1 = quaternion(q(1), q(2), q(3), q(4));
    o = quaternion(0, omega(1), omega(2), omega(3));

    % Quaternion derivative
    dq = 0.5*q1*o;
    [a0, a1, a2, a3] = dq.parts;

    % Omega derivative
    I11 = inertia(1);
    I22 = inertia(2);
    I33 = inertia(3);

    w1 = -(I33-I22)*omega(2)*omega(3)/I11;
    w2 = -(I11-I33)*omega(3)*omega(1)/I22;
    w3 = -(I22-I11)*omega(1)*omega(2)/I33;

    dqdt = [a0, a1, a2, a3, w1, w2, w3]';

end
```

Error remained under 2e-8 and the conversion from Euler Parameters to Euler Angles can be done with the codes from problem sheet 1

E 3

The code used to solve the problem is the following

```
%% Inputs
IS = 1500; % kgm²
IR = 500; % kgm²

w_0 = 1;
wr_0 = -10;
wr_1 = -9;


%% Calculations

% Angular moment
H = IS*w_0 + IR*(wr_0+w_0);

% (a) New station rotation speed
w_1 = (H-IR*wr_1)/(IS+IR);

% Old kinetic energy
T_0 = 0.5*IS*w_0^2+0.5*IR*(wr_0+w_0)^2;

% (b) New kinetic energy
T_1 = 0.5*IS*w_1^2+0.5*IR*(wr_1+w_1)^2;
```

```
loss = T_0-T_1; % J
```

```
w_1 =
    0.7500 rad/s
```

a

```
loss =
    3.5625 kJ
```

b

c For this transfer of energy to occur, some active system must actively create the internal torque, therefore consuming energy. As can be seen from the equations, the lack of external torque only implies a conservation of the angular moment, not kinetic energy. Other kind of **internal** energy must be used for the operation.

E 4

The code used to solve the problem is the following

```
%% Inputs
I_S = 80 % kg m^2
I_T = 60 % kg m^2
wz_0 = 2; % rad s^-1

a = deg2rad(15); % rad

%% Calculations

% Angular moment (remains constant cause no external torque)
Hz = I_S * wz_0;
H = Hz / cos(a);
Ht = H * sin(a);
wt_0 = Ht / I_T; % x,y rotation

% (a) Final spin rate
wz_1 = H / I_S; % rad/s

% Old kinetic energy
T_0 = 0.5*(I_T*wt_0^2 + I_S*wz_0^2);

% (b) New kinetic energy and loss
T_1 = 0.5*I_S*wz_1^2; % J
loss = T_0-T_1; % J
```

```
wz_1 =
    2.0706 rad /s
```

a

```
loss =
   3.8292 J
```

b

E 5

Using the graphical solution (**H** pointing outwards) we end up with the following equations:

$$\mathbf{L} = \frac{\Delta \mathbf{H}}{\Delta t}; \qquad \Delta \mathbf{H} = \mathbf{H}_1 - \mathbf{H}_0$$

$$\mathbf{H}_0 = H\hat{\mathbf{z}}; \qquad \mathbf{H}_1 = H\cos\theta\hat{\mathbf{z}} + H\sin\theta\hat{\mathbf{x}}$$

$$\theta = \dot{\theta}\Delta t; \quad \dot{\theta} = \frac{2\pi}{T}$$

Being a stationary orbit, T = 24h, therefore $\dot{\theta}$ becomes a very small value. If we also end with a very small value of Δt, the moment equation can be approximated to:

$$\mathbf{L} = \frac{\Delta \mathbf{H}}{\Delta t} = \frac{H\cos\theta\hat{\mathbf{z}} + H\sin\theta\hat{\mathbf{x}} - H\hat{\mathbf{z}}}{\Delta t} \approx \frac{H\hat{\mathbf{z}} + H\dot{\theta}\Delta t\hat{\mathbf{x}} - H\hat{\mathbf{z}}}{\Delta t} = H\dot{\theta}\hat{\mathbf{x}}$$

$$L = H\dot{\theta} = I_s\omega_s \cdot \frac{2\pi}{24 \cdot 60 \cdot 60} = 3.808 \text{ mN m}$$

Expressed from the orbital frame.

Using the transport theorem:

$$\left(\frac{d\mathbf{H}}{dt}\right)\text{\_inertial} = \left(\frac{d\mathbf{H}}{dt}\right)\text{\_orbital} + \boldsymbol{\omega}\text{\_orb} \times \mathbf{H}$$

Since we want the spin axis to stay fixed in the orbital frame $\left(\frac{d\mathbf{H}}{dt}\right)\text{\_orbital} = 0$ so:

$$\mathbf{L} = \boldsymbol{\omega}_{\text{orb}} \times \mathbf{H}$$

Which is the same result we got before as $\boldsymbol{\omega}_{\text{orb}}$ has the value $\dot{\theta}\hat{\mathbf{y}}$, still expressed on the orbital frame.

E 6

The code used to solve this problem is the following:

```
%% Inputs
IS = 1000; % kg m²
IG = 6000; % kg m²

ws_0 = 0.4; % rad/s
a = deg2rad(20); % rad

%% Calculations
```

Trajectory d)

Figure 1: Trajectory d)

```
H0 = IS * ws_0;
H1 = H0 / cos(a);
precession_rate = -H1/IG; % (4-77a, page 170)
t = abs(pi / precession_rate); % 180º in rad
```

Solution:

```
t =
    44.2820 s
```

C 1

a

Jacobian

$$A = \begin{pmatrix} 0 & \frac{I_2-I_3}{I_1-I_s}\omega_3 & \frac{I_2-I_3}{I_1-I_s}\omega_2 & \frac{v_d}{I_1-I_s} & 0 & 0 \\ \frac{I_3-I_1}{I_2-I_s}\omega_3 & 0 & \frac{I_3-I_1}{I_2-I_s}\omega_1 & 0 & \frac{v_d}{I_2-I_s} & 0 \\ \frac{I_1-I_2}{I_3-I_s}\omega_2 & \frac{I_1-I_2}{I_3-I_s}\omega_1 & 0 & 0 & 0 & \frac{v_d}{I_3-I_s} \\ 0 & -\sigma_3 & \sigma_2 & -\frac{v_d}{I_s} & \omega_3 & -\omega_2 \\ \sigma_3 & 0 & -\sigma_1 & -\omega_3 & -\frac{v_d}{I_s} & \omega_1 \\ -\sigma_2 & \sigma_1 & 0 & \omega_2 & -\omega_1 & -\frac{v_d}{I_s} \end{pmatrix}$$

Given the Jacobian A, we can now compute the eigen values and estimate the stabiltiy of the system on the different points.

```
eigen_a =
    0.0000 -1.6894+0.9955i -1.6894-0.9955i -0.0054 0.0000 -1.6818
```

As all real parts of the eigen values are negative, we are at a stable point.

b

```
eigen_b =
    0.0000 -1.6972 0.0014+0.4087i 0.0014-0.4087i -1.6858+1.0040i -1.6858-1.0040i
```

As the 3rd and 4th components have real positive parts, we are not at a stable point.

c

```
eigen_c =
   0.3584 -0.3584 -1.6667-1.0000i  -1.6667+1.0000i   0.0000 -1.6667
```

Similar to part b, we still have real positive eigenvalues, therefore, we are not at a stable point

Angular momentum magnitude d)

Figure 2: Angular momentum magnitude d)

Trajectory e)

Figure 3: Trajectory e)

d

Trajectory:

Angular momentum magnitude:

e

Trajectory:

Angular momentum magnitude:

Code

The code used to solve problem **d** and **e** is the following:

```
clear all; clc; close all;
addpath('./rotLib/')
%% Inputs
I1 = 2000; % kg m²
I2 = 1500; % kg m²
I3 = 1000; % kg m²
Is = 18;   % kg m²
vd = 30;   % N m s

statics = [I1 I2 I3 Is vd];

% initial_state: w1, w2, w3, sig1, sig2, sig3 rad/s
initial_state_d = [0.1224, 0, 2.99, 0, 0, 0];
initial_state_e = [0.125, 0, 2.99, 0, 0, 0];

%% Calculations

% (d)
tspan = [0 2000];
options = odeset('RelTol', 1e-8, 'AbsTol', 1e-8);
[t, c_state] = ode45(@(t, s) state_derivative(s, statics), tspan, initial_state_d, options);
% [t, c_state] = ode45(@(t, s) state_derivative(s, statics), tspan, initial_state);
a_m = @(s) angular_moment(s, statics);
result = cellfun(a_m, num2cell(c_state, 2));
figure();
plot(t, result);
end_d = c_state(end, :)

% (e)
```

Angular momentum magnitude: e)

Figure 4: Angular momentum magnitude: e)

```
tspan = [0 2000];
options = odeset('RelTol', 1e-8, 'AbsTol', 1e-8);
[t, c_state] = ode45(@(t, s) state_derivative(s, statics), tspan, initial_state_e, options);
% [t, c_state] = ode45(@(t, s) state_derivative(s, statics), tspan, initial_state);
a_m = @(s) angular_moment(s, statics);
result = cellfun(a_m, num2cell(c_state, 2));
figure();
plot(t, result);

end_e = c_state(end, :)
%% Function

function dsdt = state_derivative(state, statics)
    I1 = statics(1);
    I2 = statics(2);
    I3 = statics(3);
    Is = statics(4);
    vd = statics(5);

    w = state(1:3);
    sig = state(4:6);

    wd1 = ((I2-I3)*w(2)*w(3)+vd*sig(1))/(I1-Is);
    wd2 = ((I3-I1)*w(3)*w(1)+vd*sig(2))/(I2-Is);
    wd3 = ((I1-I2)*w(1)*w(2)+vd*sig(3))/(I3-Is);

    sigd1 = -wd1 - (vd/Is)*sig(1) - w(2)*sig(3) + w(3)*sig(2);
    sigd2 = -wd2 - (vd/Is)*sig(2) - w(3)*sig(1) + w(1)*sig(3);
    sigd3 = -wd3 - (vd/Is)*sig(3) - w(1)*sig(2) + w(2)*sig(1);

    dsdt = [wd1, wd2, wd3, sigd1, sigd2, sigd3]';

end

function H = angular_moment(state, statics)
    I1 = statics(1);
    I2 = statics(2);
    I3 = statics(3);
    Is = statics(4);

    w = state(1:3);
    sig = state(4:6);

    H2 = (I1*w(1) + Is*sig(1))^2 + (I2*w(2) + Is*sig(2))^2 + (I3*w(3) + Is*sig(3))^2;
    H = sqrt(H2);
end
```

C 2

a

```
clear all; clc; close all;
addpath('./rotLib/')
```

p2a1

Figure 5: p2a1

```
%% Inputs
% Inertia of the gimball
I_g_a = 0.1;        % kgm²
I_g_t = 0.05;       % kgm²
w_g = 1E+04;        % rad/s
I_g = [I_g_t   0     0
         0   I_g_t   0
         0     0   I_g_a];

% Inertia of the rocket
I_r_a = 1E+03;      % kgm²
I_r_t = 1E+04;      % kgm²
w_r = 10;           % rad/s
I_r = [I_r_a   0     0
         0   I_r_t   0
         0     0   I_r_t];
%% Calculations
HR = I_r * [w_r, 0 ,0]';

HG = I_g * [0, 0, w_g]';


finalH = HR + HG;
fHmag = norm(finalH);

finalI = I_r + I_g;
finalw = (finalI^-1)*finalH;

% Asuming I2 = I3
precession_rate = -fHmag/I_r_t; % (4-77a, page 170)
time = 2*pi/precession_rate;

% max angle  % (4-79, page 171)
theta = asin(-(finalI(1,1)*finalw(1))/(finalI(2,2)*precession_rate));
max_angle = pi/2 - theta;

% Since the rocket axis of symmetry is rotating around H
% the maximum angular deviation is:
max_angle*2
```

```
time =
    6.2520  % s
```

```
max_angle*2 =
    0.1994 % rad
```

Sketch of the spacecraft axis

Figure 6: Sketch of the spacecraft axis

Energy validation

Figure 7: Energy validation

A 1

Calculation

Sketch of the spacecraft axis

Energy validation

Comment

The significant decrease of axial inertia compared to the transverse one would have implied a significant increment in the angle. However, due to the spending of internal energy to convert it into kinetic energy, the angle was decreased as these devices usually intend.

A 2

a

Defining θ as the angle between ω and the axis of symmetry $\hat{i}$:

$$\mathbf{H} = H_a\hat{i} + H_t\hat{t}\ ; \qquad \boldsymbol{\omega} = \omega_a\hat{i} + \omega_t\hat{t}\ ; \qquad \omega_a = w\cos\theta\ ; \quad w_t = w\sin\theta$$

$$I_a = kI_t; \qquad H_a = I_a\omega_a = kI_t\omega_a; \quad H_t = I_t\omega_t$$

Now, using the dot product we can find the angle between $\mathbf{H}$ and $\boldsymbol{\omega}$.

$$\cos\beta = \frac{\mathbf{H}\cdot\boldsymbol{\omega}}{|\mathbf{H}||\boldsymbol{\omega}|}$$

The maximum value happens at the minimum value of the content inside, given f:

$$f(\theta) = \frac{k\cdot\cos^2(\theta) + sin^2(\theta)}{\sqrt{k^2\cdot\cos^2(\theta) + sin^2(\theta)}}$$

$$f'(\theta) = \frac{(k-1)^2\cos(\theta)\sin(\theta)\left(\sin^2(\theta) - k\cos^2(\theta)\right)}{\left(\sin^2(\theta) + k^2\cos^2(\theta)\right)^{\frac{3}{2}}}$$

Which is 0 for the $\theta = 0.886077$rad, getting us $\beta = 11.5371$

b

We can know this maximum exists as the f' is positive and $f = 0$ at $\theta = 0$ and f' is positive and $f = 0$ at $\theta = \frac{\pi}{2}$. As both functions are continuous. We know that there is a point in the middle where f' is 0 and f'' needs to be negative. The range of values of θ goes from 0 to $\frac{\pi}{2}$ as higher values are equivalent in terms of symmetry.

c

The kinetic energy at the largest value of β is given by:

$$T = \frac{1}{2}(I_t\omega_t^2 + I_a\omega_a^2) = \frac{1}{2}(\frac{H_t^2}{I_t} + \frac{H_a^2}{I_a}) = \frac{1}{2}(\frac{H_t^2}{I_t} + \frac{H_a^2}{kI_t})$$

$$H_a = H\cos(\theta - \beta); \qquad H_t = H\sin(\theta - \beta);$$

$$T = \frac{H^2}{2I_t}(\sin^2(\theta - \beta) + \frac{\cos^2(\theta - \beta)}{k})$$

$$T_{\text{max}_\text{a}\text{ngle}} = 0.8000\frac{H^2}{2I_t}$$

$$T_{\text{max}} = \frac{H^2}{2I_t}$$

$$T_{\text{min}} = \frac{H^2}{2I_a} = \frac{H^2}{3I_t}$$

The kinetic energy is well within the boundaries.

$$T_{\text{min}} < T_{\text{max angle}} < T_{\text{max}}$$