Table of Contents


```
%-------- HW 8 MATLAB code --------%
% Romeo Perlstein, section 0101    %
% Chat it's over... I'm cooked!!!  %
close all
```

Q1

a

given the 3-2-1 euler rotation of a body:

```
theta1= 30;
theta2 = 40;
theta3 = 10;
Tz = [cosd(theta1), -sind(theta1), 0; sind(theta1), cosd(theta1), 0; 0,0,1];
Ty = [cosd(theta2), 0, sind(theta2); 0, 1, 0; -sind(theta2), 0, cosd(theta2)];
Tx = [1,0,0; 0, cosd(theta3), -sind(theta3); 0, sind(theta3), cosd(theta3)];

R_full = Tz*Ty*Tx
```

b

Get the principle rotation angle

```
phi = acosd(.5*(R_full(1,1)+R_full(2,2)+R_full(3,3)-1));

% Get the principle axis
e_vec = 1/(2*sind(phi)) * [R_full(2,3)-R_full(3,2);R_full(3,1)-
R_full(1,3);R_full(1,2)-R_full(2,1)]
```

c

Now, find the quaternion values:

```
q1 = e_vec(1)*sind(phi/2)
q2 = e_vec(2)*sind(phi/2)
q3 = e_vec(3)*sind(phi/2)
q4 = cosd(phi/2)
if((q1^2 + q2^2 + q3^2 + q4^2) > .99999 && (q1^2 + q2^2 + q3^2 + q4^2) <
 1.00001)
    fprintf("Quaternions check out!\n\n");
end
```

d

get the quaternion velocity

```
w_vec = [0;0.1;0.2;0];
B_vec = [q4;q1;q2;q3];
B_TRANSFORMATION = [B_vec(1), -B_vec(2), -B_vec(3)  -B_vec(4);
                    B_vec(2),  B_vec(1), -B_vec(4),  B_vec(3);
                    B_vec(3),  B_vec(4),  B_vec(1), -B_vec(2);
                    B_vec(4), -B_vec(3),  B_vec(2),  B_vec(1)];
B_dot_vec = (1/2)*B_TRANSFORMATION*w_vec
```

```
R_full =

    0.6634   -0.3957    0.6350
    0.3830    0.9087    0.1661
   -0.6428    0.1330    0.7544


e_vec =

    0.0221
   -0.8537
   -0.5203


q1 =

    0.0091


q2 =

   -0.3503


q3 =

   -0.2135


q4 =
```

```
    0.9119

Quaternions check out!


B_dot_vec =

    0.0346
    0.0669
    0.0805
    0.0184
```

Q2

Considering a torque-free rigid body:

```
I = [10,  0,  0;
      0, 20,  0;
      0,  0, 30];
```

a

given the angular velocity vector, find angular momentum at that point and kinetic energy

```
w_vec_deg = [10;0;30];
w_vec = w_vec_deg * (pi/180);

h_vec = [I(1,1)*w_vec(1); I(2,2)*w_vec(2); I(3,3)*w_vec(3);]
h_vec_norm = norm(h_vec);
KE = .5*I(1,1)*(w_vec(1))^2 + .5*I(2,2)*(w_vec(2))^2 + .5*I(3,3)*(w_vec(3))^2
```

b

use the ODE propagator to find the angular velocity over time

```
%--- ODE func values---%
tall_er_ant = (10^-13); % Tolerance
step_size = 1; % step size
max_time = 100; % max time (0->max_time)
t = [0:step_size:max_time]; % timestep

% ODE options
ODE_options = odeset("RelTol", tall_er_ant, "AbsTol", tall_er_ant);

[T1, Y1] = ode45(@myodefun, t, w_vec, ODE_options, I);
hold on
plot(T1, Y1(:,1), DisplayName="\omega_x")
plot(T1, Y1(:,2), DisplayName="\omega_y")
plot(T1, Y1(:,3), DisplayName="\omega_z")
title("Plot of angular velocity of a torqe-free rigid body")
```

```
xlabel("Time (seconds)")
ylabel("Angular Velocity (rads/sec)")
legend
grid on
```

c

```
KE_vec_C = .5*I(1,1).*(Y1(:,1)).^2 + .5*I(2,2).*(Y1(:,2)).^2
 + .5*I(3,3).*(Y1(:,3)).^2;

for i=1:1:length(T1)
    h_norm_C(i) = norm([I(1,1).*Y1(i,1);I(2,2).*Y1(i,2);I(3,3).*Y1(i,3);]);
    h_vec_array(i,1:3) = [I(1,1).*Y1(i,1);I(2,2).*Y1(i,2);I(3,3).*Y1(i,3)];
end
figure
tiledlayout(1,2)
nexttile
plot(T1, KE_vec_C);
title("Plot of Kinetic Energy deviations over time (zoomed in)")
xlabel("Time (seconds)")
ylabel("Kinetic Energy (Joules)")
nexttile
plot(T1, KE_vec_C);
title("Plot of Kinetic Energy deviations over time (equal axis)")
xlabel("Time (seconds)")
ylabel("Kinetic Energy (Joules)")
axis equal

figure
tiledlayout(1,2)
nexttile
plot(T1, h_norm_C);
title("Plot of Angular Momentum over time (zoomed in)")
xlabel("Time (seconds)")
ylabel("Angular Momentum (kg-m^2/s")
nexttile
plot(T1, h_norm_C);
title("Plot of Angular Momentum over time (equal axis)")
xlabel("Time (seconds)")
ylabel("Angular Momentum (kg-m^2/s")
axis equal

% I believe my code is working because my Kinetic Energy and my Angular
% momentum magnitude is constant! This is inline with what is expected when
% assuming a torque-free rigid body, that the angular momentum and energy
% should remain constant as it spins (because there are no external forces
% acting on it, it shouldn't lose energy or momentum!)
```

d

```
% Create the sphere:
% Get our h squared value
```

```
h_squared = I(1,1)^2*w_vec(1)^2 + I(2,2)^2*w_vec(2)^2 + I(3,3)^2*w_vec(3)^2;
x = sqrt(h_squared); % since h_squared = r^2, sqrt(h_squared) = r, our radius!
y = sqrt(h_squared);
z = sqrt(h_squared);
[theta,phi] = ndgrid(linspace(0,pi),linspace(0,2*pi));
X = x*sin(theta).*cos(phi);
Y = y*sin(theta).*sin(phi);
Z = z*cos(theta);

% Next, create the ellipsoid!
a = sqrt((2*I(1,1)*KE)); % implement the equation from the lecture
b = sqrt((2*I(2,2)*KE)); % since b^2 = (2*I(2,2)*KE, b = sqrt((2*I(2,2)*KE)
c = sqrt((2*I(3,3)*KE)); % do the same for c^2, a^2 as above for b^2
[theta,phi] = ndgrid(linspace(0,pi),linspace(0,2*pi));
A = a*sin(theta).*cos(phi);
B = b*sin(theta).*sin(phi);
C = c*cos(theta);

% Finally, plot the sucker!
figure
hold on
surf(X,Y,Z, FaceColor="cyan", EdgeColor="blue", DisplayName="Angular Momentum
 Sphere");
surf(A,B,C, FaceColor="cyan", EdgeColor="blue", DisplayName="Kinetic Energy
 Ellipsoid")
title("Polhode plot of the initial \omega vector, h, and KE ")

scatter3(h_vec_array(:,1), h_vec_array(:,2),
 h_vec_array(:,3), "red", "filled", DisplayName="Plot of time-varying angular
 moment")
legend
axis equal
```

e

```
w_vec_deg2 = [1;15;0];
w_vec2 = w_vec_deg2*(pi/180);
[T2, Y2] = ode45(@myodefun, t, w_vec2, ODE_options, I);
figure
hold on
plot(T2, Y2(:,1), DisplayName="\omega_x")
plot(T2, Y2(:,2), DisplayName="\omega_y")
plot(T2, Y2(:,3), DisplayName="\omega_z")
title("Plot of angular velocity of a torqe-free rigid body (part f)")
xlabel("Time (seconds)")
ylabel("Angular Velocity (rads/sec)")
legend
grid on
```

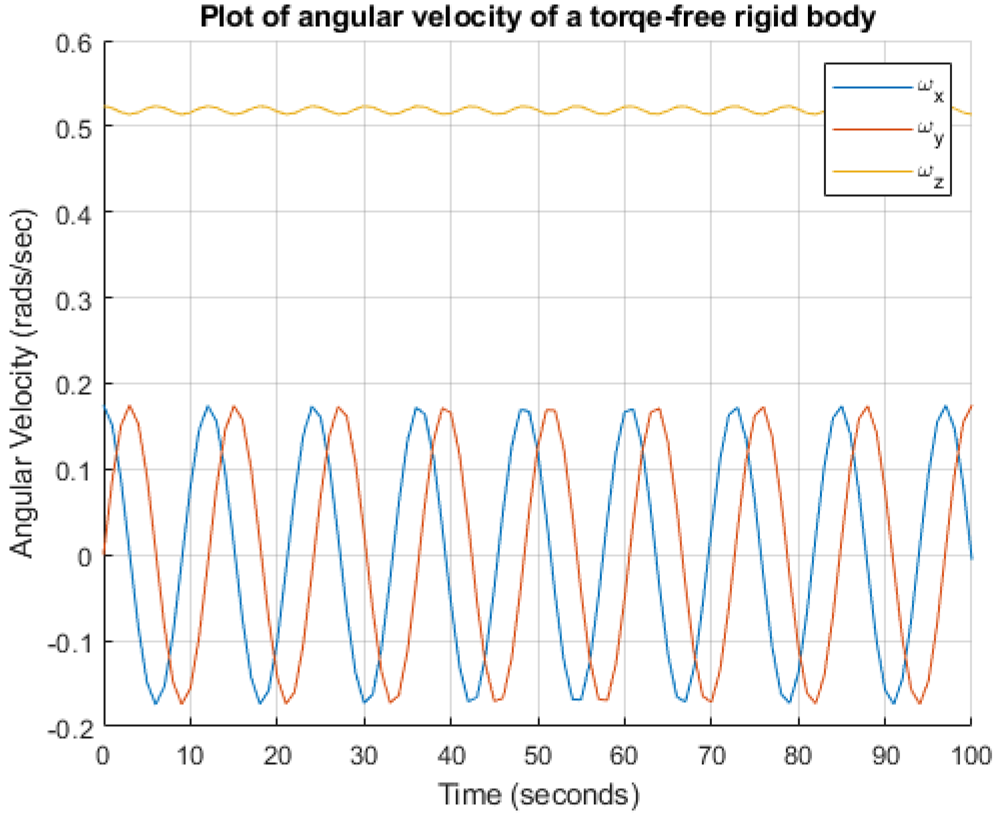
f

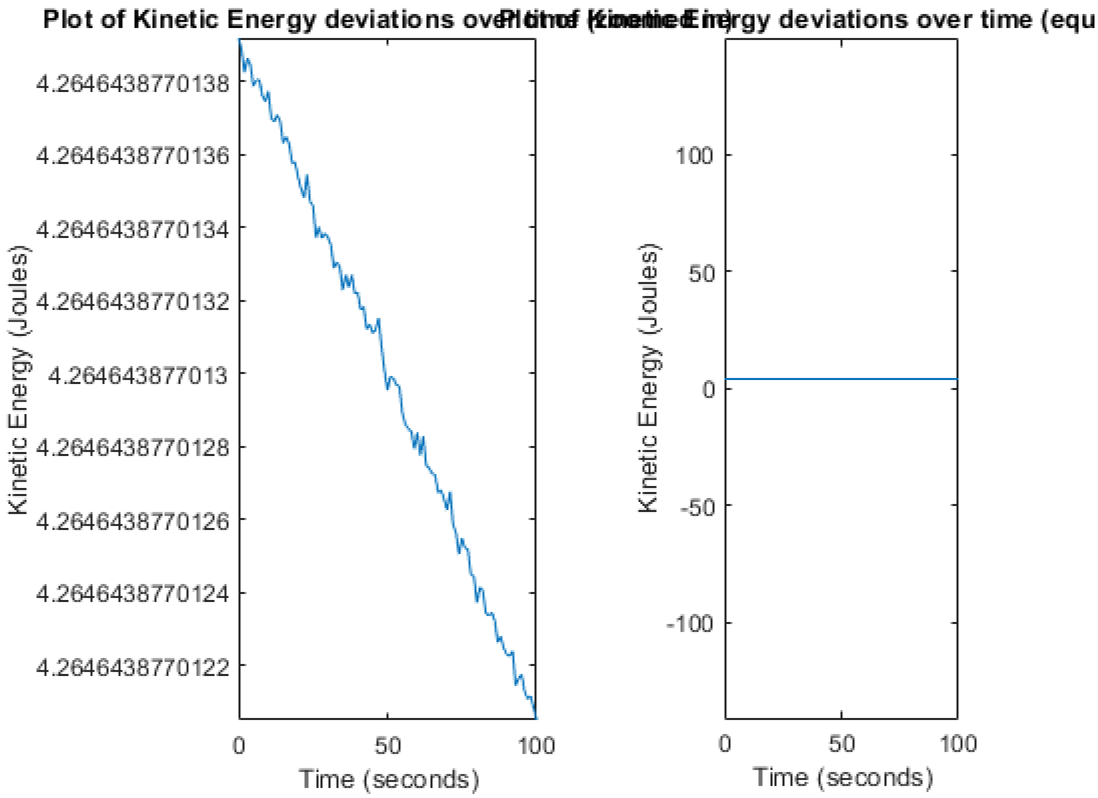
Pretty much copy/paste from c and d Get kinetic Energy

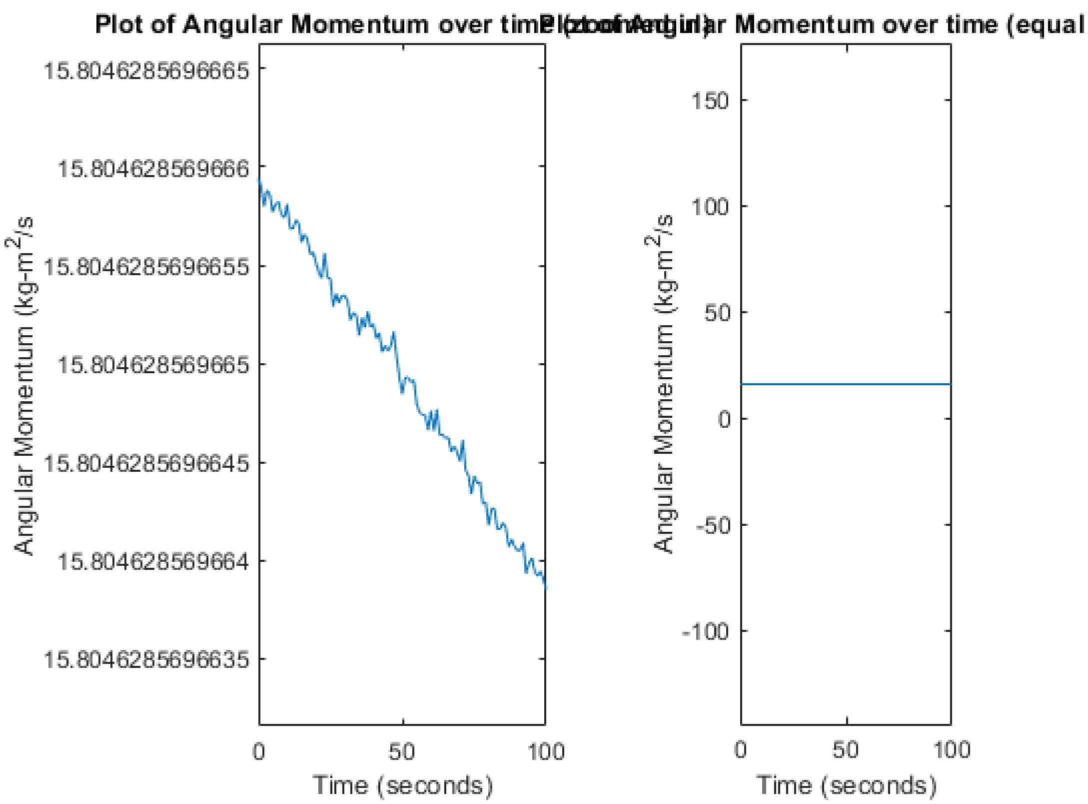
```
KE2 = .5*I(1,1)*(w_vec2(1))^2 + .5*I(2,2)*(w_vec2(2))^2
 + .5*I(3,3)*(w_vec2(3))^2;

% Get Angular momentum
for i=1:1:length(T2)
    h_vec_array2(i,1:3) = [I(1,1).*Y2(i,1);I(2,2).*Y2(i,2);I(3,3).*Y2(i,3)];
end

% Get our h squared value
h_squared2 = I(1,1)^2*w_vec2(1)^2 + I(2,2)^2*w_vec2(2)^2 +
 I(3,3)^2*w_vec2(3)^2;
x = sqrt(h_squared2); % since h_squared = r^2, sqrt(h_squared) = r, our
 radius!
y = sqrt(h_squared2);
z = sqrt(h_squared2);
[theta,phi] = ndgrid(linspace(0,pi),linspace(0,2*pi));
X = x*sin(theta).*cos(phi);
Y = y*sin(theta).*sin(phi);
Z = z*cos(theta);

% Next, create the ellipsoid!
a = sqrt((2*I(1,1)*KE2)); % implement the equation from the lecture
b = sqrt((2*I(2,2)*KE2)); % since b^2 = (2*I(2,2)*KE, b = sqrt((2*I(2,2)*KE)
c = sqrt((2*I(3,3)*KE2)); % do the same for c^2, a^2 as above for b^2
[theta,phi] = ndgrid(linspace(0,pi),linspace(0,2*pi));
A = a*sin(theta).*cos(phi);
B = b*sin(theta).*sin(phi);
C = c*cos(theta);

% Finally, plot the sucker!
figure
hold on
surf(X,Y,Z, FaceColor="cyan", EdgeColor="blue", DisplayName="Angular Momentum
 Sphere");
surf(A,B,C, FaceColor="cyan", EdgeColor="blue", DisplayName="Kinetic Energy
 Ellipsoid")
title("Polhode plot of the \omega vector from part f), h, and KE ")

scatter3(h_vec_array2(:,1), h_vec_array2(:,2),
 h_vec_array2(:,3), "red", "filled", DisplayName="Plot of time-varying angular
 moment")
legend
axis equal

% I think the plots from b and d look different because our angular
% velocity is about a completely different axis. from the first half of the
% HW, our angular velocity was about x and z, but now we are rotation about
% x any y, meaning our motion will be completely different. Also, our
% angular velocities are quite different, since our rotation about x is
% much smaller than the first half of the problem (1 << 15, compared to 10
% and 30 from the first half).
```

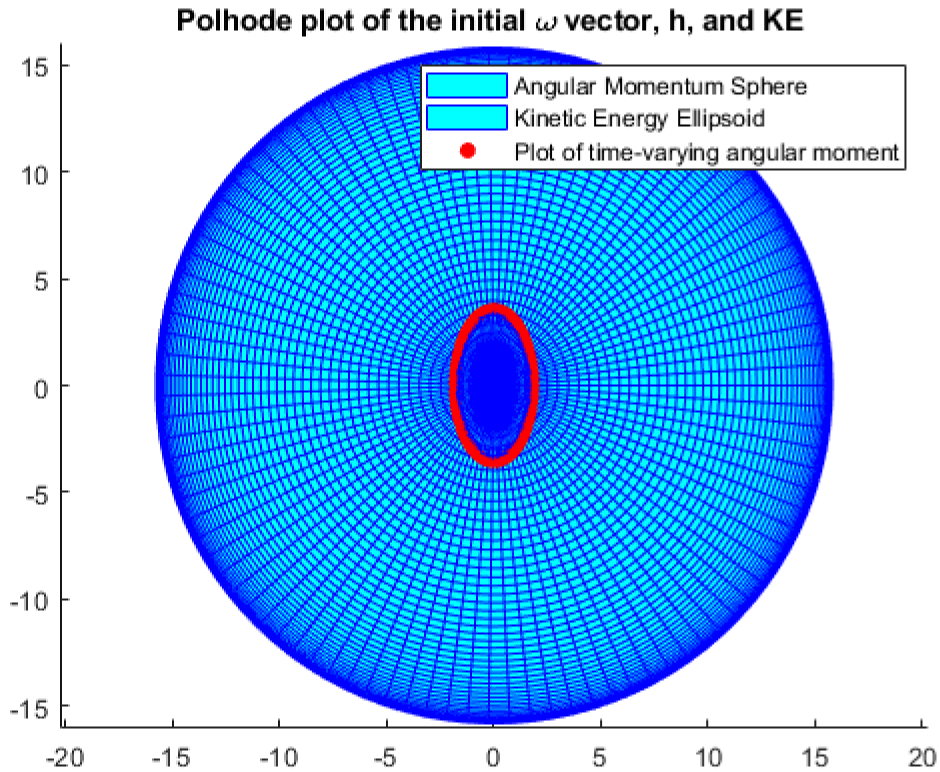
```
% 2-body prop solver, pulled from ENAE301! Ah good times... good times.
function ydot = myodefun(t, y, I)
    ydot(1,1) = ( -(I(3,3)-I(2,2))*y(2)*y(3) )/(I(1,1));
    ydot(2,1) = ( -(I(1,1)-I(3,3))*y(3)*y(1) )/(I(2,2));
    ydot(3,1) = ( -(I(2,2)-I(1,1))*y(1)*y(2) )/(I(3,3));
end
```

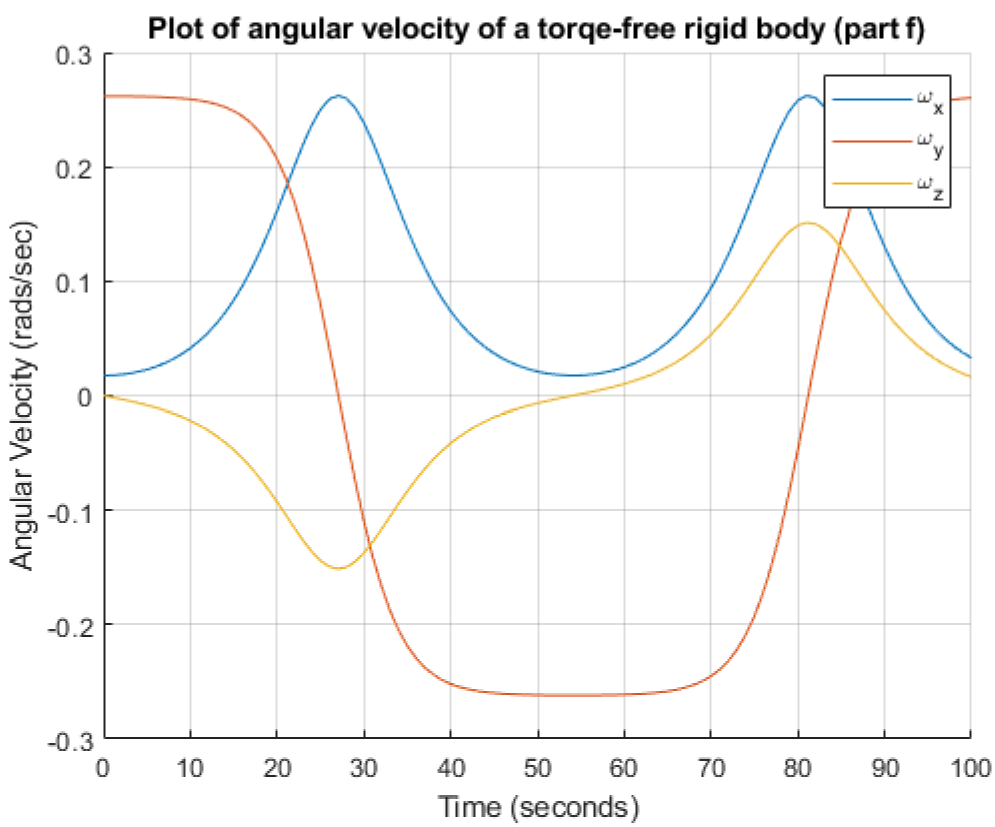
```
h_vec =

    1.7453
         0
   15.7080


KE =

    4.2646
```

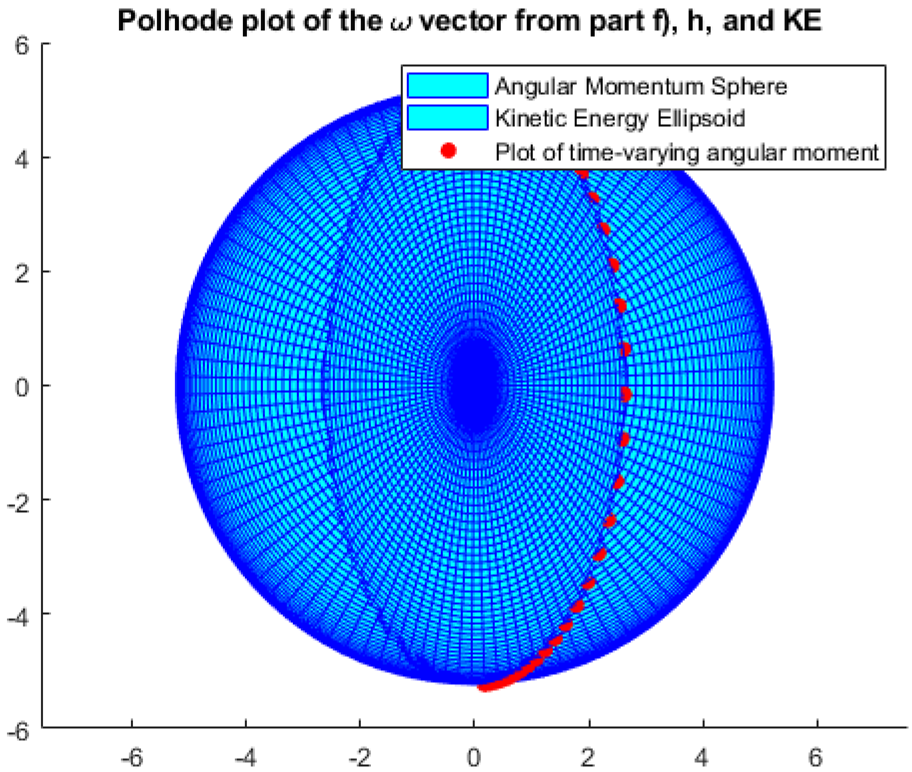



Plot of Kinetic Energy deviations ove
rgy deviations over time (equ
Kinetic Energy (Joules)
4.2646438770138
4.2646438770136
4.2646438770134
4.2646438770132
4.264643877013
4.2646438770128
4.2646438770126
4.2646438770124
4.2646438770122
0
50
100
Time (seconds)
100
50
0
-50
-100


Plot of Angular Momentum over tim
r Momentum over time (equal
Angular Momentum (kg-m^2/s
15.8046285696665
15.804628569666
15.8046285696655
15.804628569665
15.8046285696645
15.804628569664
15.8046285696635
0
50
100
Time (seconds)
150
100
50
0
-50
-100


Polhode plot of the initial ω vector, h, and KE
Angular Momentum Sphere
Kinetic Energy Ellipsoid
Plot of time-varying angular moment


Plot of angular velocity of a torqe-free rigid body (part f)
Angular Velocity (rads/sec)
Time (seconds)
ω_x
ω_y
ω_z



Published with MATLAB® R2022b