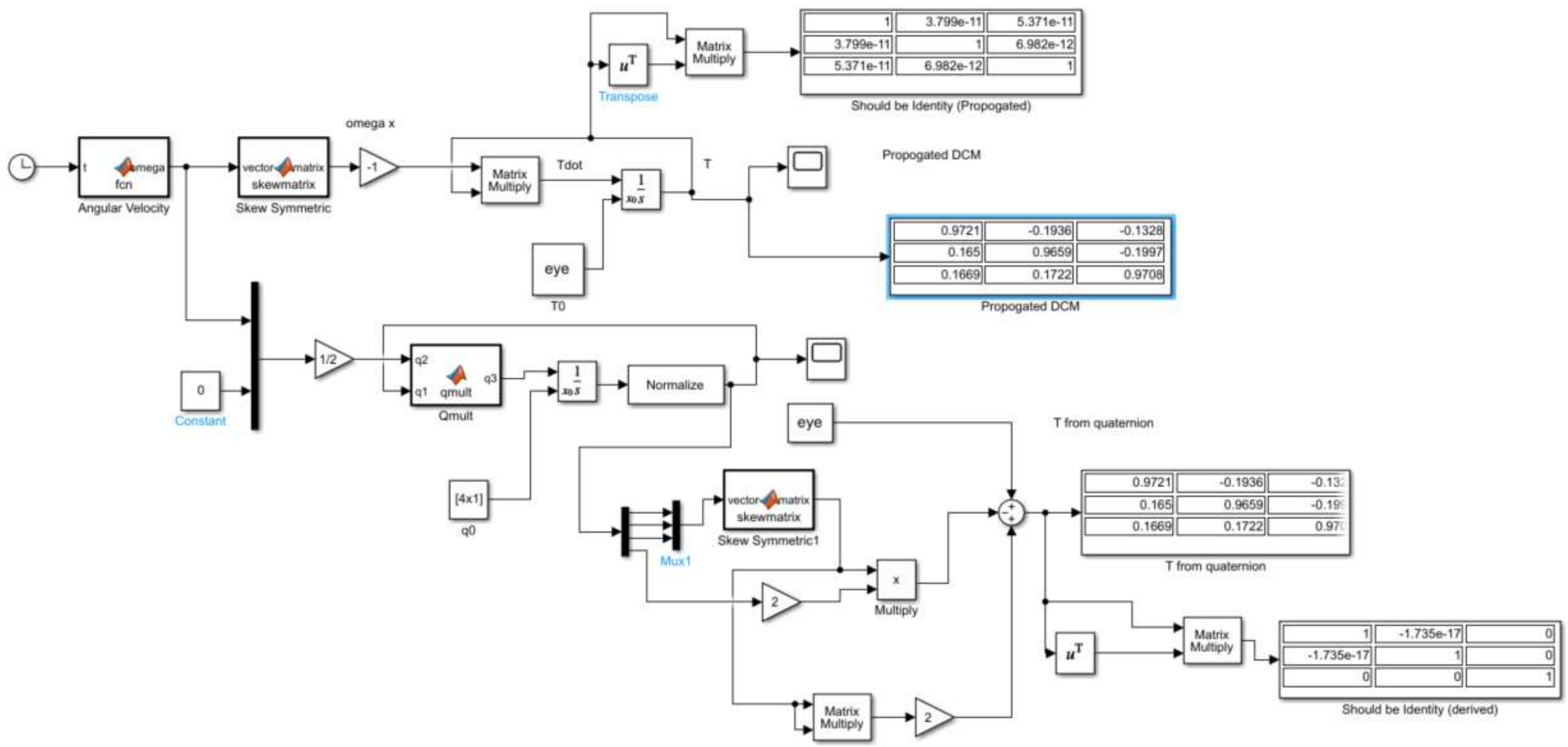
Izaac Facundo

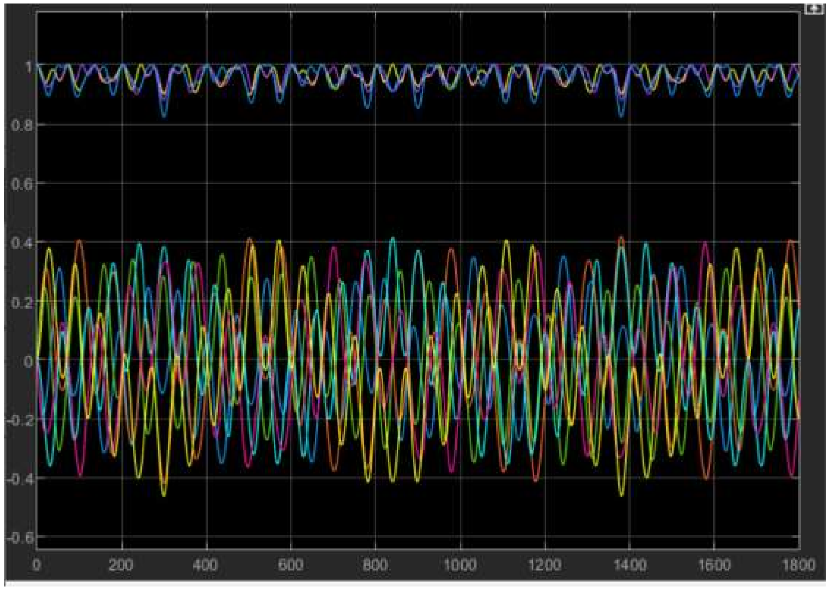
Imf339

HW4

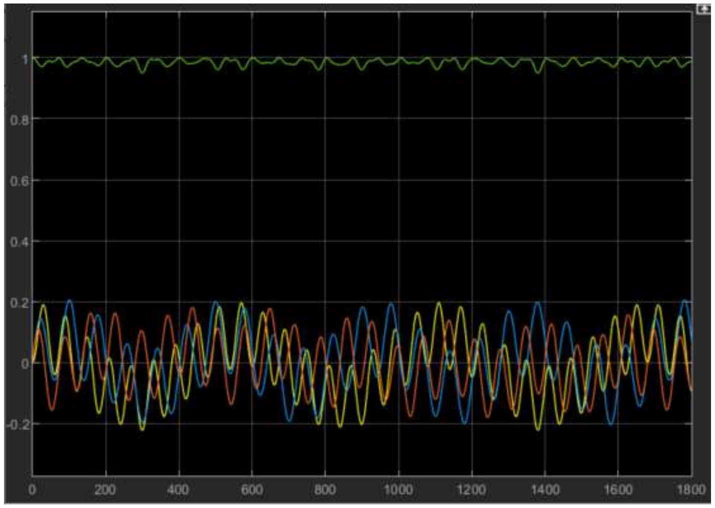
Problem 3



Plot the components of the DCM versus time



• Plot the components of the quaternion versus time



• Using the formula we saw in class, calculate the DCM from the quaternion at time t = 1800 seconds and make sure it (approximately) matches the propagated DCM at time t = 1800 seconds

0.9721	-0.1936	-0.1328
0.165	0.9659	-0.1997
0.1669	0.1722	0.9708

Propogated DCM

0.9721	-0.1936	-0.1328
0.165	0.9659	-0.1997
0.1669	0.1722	0.9708

T from quaternion

• Using the propagated DCM show the value of T T T at the final time (hint: they should be close to the identity matrix).

1	3.799e-11	5.371e-11
3.799e-11	1	6.982e-12
5.371e-11	6.982e-12	1

Should be Identity (Propogated)

• Using the DCM calculated from the quaternion show the values of T T T at the final time (hint: they should be close to the identity matrix).

1	-5.365e-12	6.419e-12
-5.365e-12	1	-5.173e-12
6.419e-12	-5.173e-12	1

Should be Identity (derived from quaternion)

• Which of the two is closer to identity?

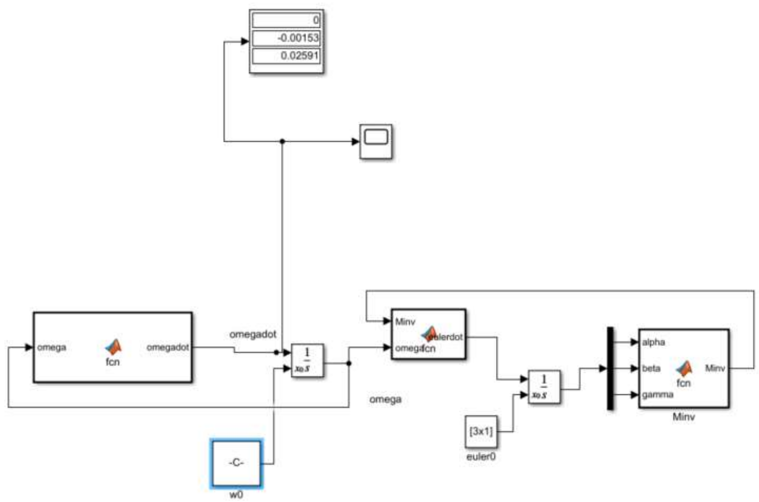
The one calculated from the quaternion is closer to identity

• Add a “Normalize” block after the quaternion integration, show that using the corresponding DCM at the final time we now have that T T T = I (within machine precision).

1	-1.735e-17	0
-1.735e-17	1	0
0	0	1

Should be Identity (derived from quaternion)

Problem 4:



```
function omegadot = fcn(omega)
J = [100 0 0;
    0 60 0;
    0 0 60];
omegadot = -inv(J)*[cross(omega,J*omega)];
```

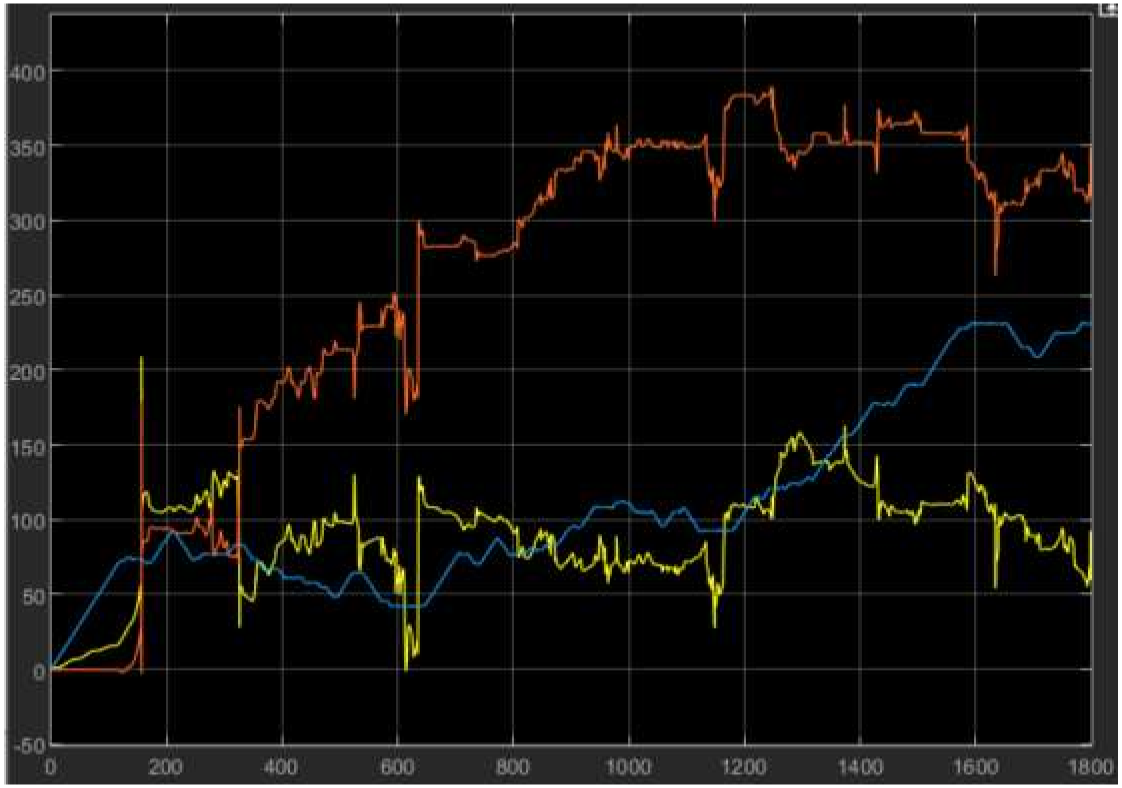
```
function eulerdot = fcn(Minv,omega)

eulerdot = Minv*omega;
```

```
function Minv = fcn(alpha,beta,gamma)

Minv = (1/cos(beta))*[sin(gamma) cos(gamma) 0;
    cos(beta)*cos(gamma) cos(beta)*sin(gamma) 0;
    -sin(beta)*sin(gamma) -sin(beta)*cos(gamma) cos(beta)];
```

I tried for a few hours but I could not get this code to produce reasonable euler angle values. Here is what I got (graphically):



400
350
300
250
200
150
100
50
0
-50
0
200
400
600
800
1000
1200
1400
1600
1800