

---

## Table of Contents

.....	1
a) .....	1
Plot the results of the controller. ....	2
b) .....	4
c) .....	6

```
close all;
clear all;

load('sat.mat')
```

**a)**

```
% First ensure that the system is controllable and observable.
cont_r = rank(ctrb(A,B)); % The rank of this is 10. So not
    controllable.
obsv_r = rank(observ(A,C)); % The rank of this is 12. So not
    observable.

% Since the system is not controllable or observable. I need to ensure
    that
% it is stabilizable and detectable.

% Detectability: The system is detectable iff every eigen value vector
    of A
% corresponding to an eigenvalue with a positive or zero real part is
    not
% in the kernal of C.
eig_A = eig(A);
detectable = 1;
for ii=1:length(eig_A)
    if real(eig_A(ii)) >= 0
        PHB = [A-eig_A(ii)*eye(18); C];
        if rank(PHB) < 18
            detectable = 0;
        end
    end
end
% The value of detectable is still 1. So the system is detectable.

% Stabilizability: The system is stabilizable iff every eigenvector of
    A'
% corresponding to and eigenvalue with a positive or zero real part is
    not
% in the kernal of B'.
eig_A = eig(A);
stabilizable = 1;
for ii=1:length(eig_A)
    if real(eig_A(ii)) >= 0
```

---

```

        PHB = [A-eig_A(ii)*eye(18) B];
        if rank(PHB) < 18
            stabilizable = 0;
        end
    end
end
% The value of stabilizable is still 1. So the system is stabilizable.

%%%% Now that I know that the system is both stabilizable and
detectable, I
%%%% can start to create an LQR and LQG controller.

%%%% LQR
% I will use Bryson's rule to create my Q1 and R1 matrices as a
starting point for
% the LQR controller.
max_x = 2; % Desired maximum state deviation from
equilibrium.
Q1 = diag(ones(18,1)/max_x^2);
q = 1; % Used to tune performance
Q = q*Q1;

max_u = 1; % Desired maximum input deviation from
equilibrium.
R1 = diag(ones(3,1)/max_u^2);
r = diag([5,2000,120]); % Used to tune performance.
R=r*R1;

[K,S,E] = lqr(A,B,Q,R);
K
eig(A-B*K); % Ensure that the eigen values are all less than zero.

%%%% LQG
% I will use the known noise covariance to and model disturbance
covariance
% To construct my D and N covariance matrices.
Dk = eye(3,3)*10^-3;
Nk = diag([ones(3,1)*10^-5;ones(3,1)*10^-10]);

[Kest,L,P] = kalman(ss(A,B,C,D),Dk,Nk);
L
eig(A-L*C); % Ensure that the eigen values are all less than zero.

```

## Plot the results of the controller.

```

figure(1),clf;
initial(ss(A-B*K,B*0,C,D),ones(18,1));

```

$K =$

*Columns 1 through 7*

---

-0.1941	0.0942	0.0588	0.0000	-0.0000	0.0000	0.0001
0.0053	0.0095	0.0024	-0.0000	-0.0000	-0.0000	0.0000
0.0062	-0.0141	0.0430	-0.0000	-0.0001	-0.0000	-0.0000

Columns 8 through 14

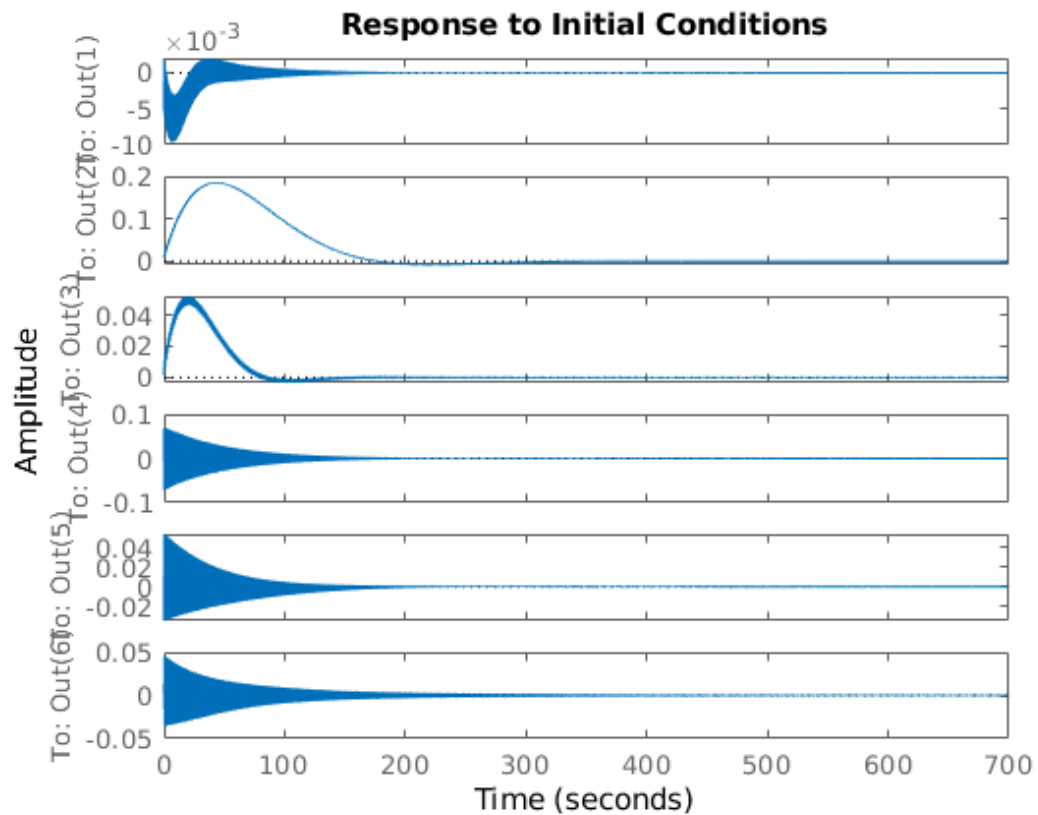
0.0000	-0.0040	-2.1812	1.0505	0.6543	0.0000	-0.0002
-0.0000	-0.0000	0.3020	0.5384	0.1328	-0.0000	-0.0000
-0.0000	-0.0000	0.1656	-0.3794	1.1516	-0.0000	-0.0014

Columns 15 through 18

0.0000	0.0004	0.0001	-0.0221
-0.0000	0.0000	-0.0000	-0.0000
-0.0000	-0.0000	-0.0004	-0.0000

$L =$

-0.3430	0.2387	0.0616	-108.2513	75.2563	19.4366
0.1584	0.4254	-0.1431	50.0020	134.1480	-45.1310
0.0992	0.1048	0.4312	31.3024	33.0538	135.9677
-0.0000	-0.0000	-0.0000	0.0019	-0.0043	-0.0063
-0.0000	-0.0000	-0.0000	-0.0397	-0.0152	-0.2099
0.0000	0.0000	-0.0000	0.0000	-0.0001	-0.0007
0.0000	0.0000	-0.0000	-0.0003	0.0433	-0.0002
0.0000	0.0000	-0.0000	0.0000	-0.0004	-0.0711
-0.0000	0.0000	-0.0000	-0.0491	0.0004	-0.0026
-0.0011	0.0008	0.0002	-196.6251	81.7363	25.3544
0.0005	0.0013	-0.0005	95.5744	152.5872	-59.0762
0.0003	0.0003	0.0014	59.8239	38.9626	182.0306
-0.0000	0.0000	0.0000	0.0226	0.0040	-1.4165
0.0000	0.0000	0.0000	0.5664	0.2701	-47.2445
0.0000	0.0000	0.0000	0.0014	-0.0651	-0.3312
0.0000	-0.0000	0.0000	1.6415	53.3897	0.0375
0.0000	0.0000	0.0000	-0.0964	-0.6793	-31.9242
0.0000	0.0000	-0.0000	-96.4643	-2.9768	-0.7414



**b)**

Run the simulation again to get the state values

```
[Y,T,X] =initial(ss(A-B*K,B*0,C,D),ones(18,1));

% Get the input values since u = -Kx;
u =-K*X';

% Plot the input
figure(2),clf;
subplot(3,1,1)
plot(T,u(1,:));
hold on
plot(T,ones(1,length(T))*1,'r')
plot(T,ones(1,length(T))*0.9,'g')
plot(T,-ones(1,length(T))*1,'r')
plot(T,-ones(1,length(T))*0.9,'g')
title('Torque about Roll');
xlabel('time (s)')
ylabel('Input Magnitude');
legend('Input','Limit','Minimum Control Effort');

subplot(3,1,2)
```

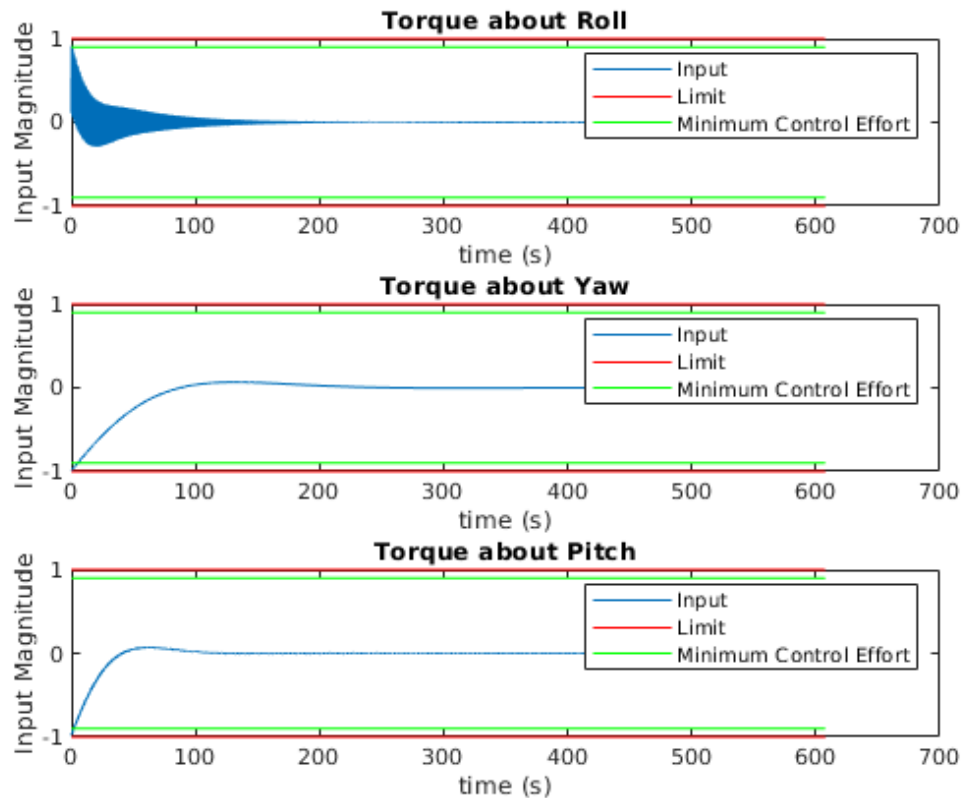
---

```
plot(T,u(2,:));
hold on
plot(T,ones(1,length(T))*1,'r')
plot(T,ones(1,length(T))*0.9,'g')
plot(T,-ones(1,length(T))*1,'r')
plot(T,-ones(1,length(T))*0.9,'g')
title('Torque about Yaw');
xlabel('time (s)')
ylabel('Input Magnitude');
legend('Input','Limit','Minimum Control Effort');

subplot(3,1,3)
plot(T,u(3,:));
hold on
plot(T,ones(1,length(T))*1,'r')
plot(T,ones(1,length(T))*0.9,'g')
plot(T,-ones(1,length(T))*1,'r')
plot(T,-ones(1,length(T))*0.9,'g')
title('Torque about Pitch');
xlabel('time (s)')
ylabel('Input Magnitude');
legend('Input','Limit','Minimum Control Effort');

% Verify that the maximum input value is between 0.9 and 1.
max(abs(u),[],2);
% The maximum values are [0.91;0.9905;0.9813]. Thus the system
  constraints
% are satisfied.
```

---



c)

```
% I want to controll yaw to 1 radian. I need to find an equilibrium
input
% for this state such that  $Ax_d + Bu_{eq} = 0$ .
x_d = zeros(18,1);
x_d(1) = 1;           % Desired x_state.
A*x_d;                % Since this value is zero, u_eq is also zero.
This                  % Simplifies the problem.

% Using section 23.6.1 in the book. I need my input to be
%  $u = -K(x-x_d) + u_{eq} \Rightarrow -K(x-x_d)$ 

temp = zeros(18,1);
temp(1) = 1;
% Set it up such that  $\dot{x} = Ax - B*K(x-x_d)$  with a step input.
[Y,T,X] = step(ss(A-B*K,+B*K*temp,C,0));

% Plot the results. I am plotting X instead of Y because Y is not even
% close to being the pointing angles and angular velocities.
figure(3),clf;
subplot(6,1,1);
```

---

```
plot(T,X(:,1))
title("Roll");
xlabel('time (s)')
ylabel('Radians');

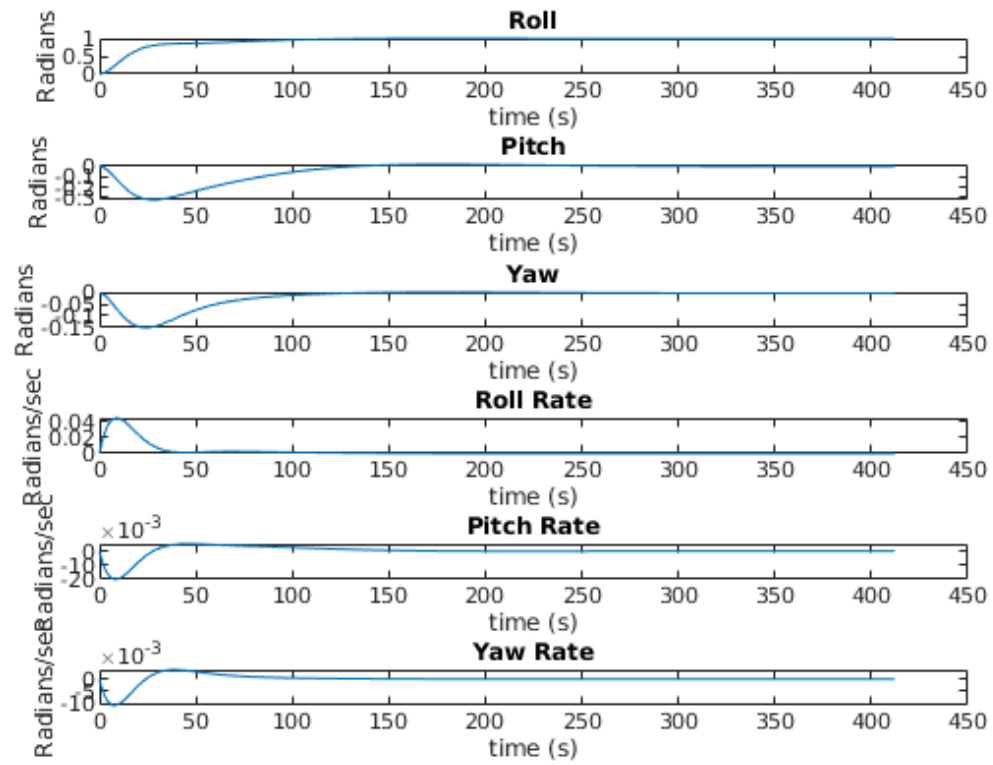
subplot(6,1,2);
plot(T,X(:,2))
title("Pitch");
xlabel('time (s)')
ylabel('Radians');

subplot(6,1,3);
plot(T,X(:,3))
title("Yaw");
xlabel('time (s)')
ylabel('Radians');

subplot(6,1,4);
plot(T,X(:,10))
title("Roll Rate");
xlabel('time (s)')
ylabel('Radians/sec');

subplot(6,1,5);
plot(T,X(:,11))
title("Pitch Rate");
xlabel('time (s)')
ylabel('Radians/sec');

subplot(6,1,6);
plot(T,X(:,12))
title("Yaw Rate");
xlabel('time (s)')
ylabel('Radians/sec');
```



*Published with MATLAB® R2018a*