AE5222 HW4

Nathaniel Goldfarb Nuttaworn Sujumnong

May 1, 2017

May 1, 2017

Problem 1

Problem 1

$$\begin{bmatrix} x \\ y \\ z \\ v \\ \psi \\ \gamma \\ T \\ L \\ \phi \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ u_1 \\ u_2 \\ u_3 \end{bmatrix} \tag{1}$$

The first step did not use the normilized constraints. This model threw errors but was produced a smooth graph that went from (0,0-100) to (2000,50). equality constraints

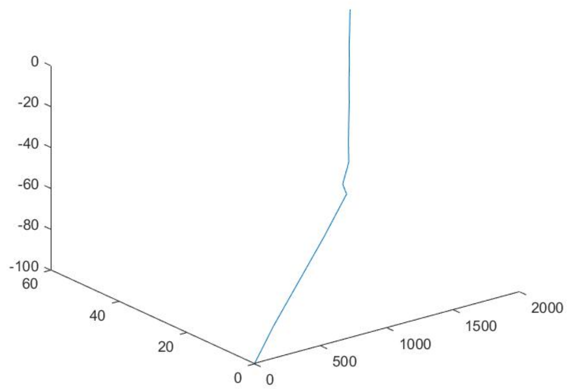
$$\begin{bmatrix} Da_1 - \frac{t_f}{2}(x_4cos(x_4)cos(x_45)) \\ Da_2 - \frac{t_f}{2}(x_4sin(x_4)cos(x_45)) \\ Da_3 - \frac{t_f}{2}(-x_4sin(x_5) \\ Da_4\frac{t_f}{2}(-\frac{1}{m}(u_1 - D - mgsin(x_6)) \\ Da_5\frac{t_f}{2}\frac{u_2sin(u_3)}{mx_4cos(x_6)} \\ Da_6\frac{1}{mx_4}(u_2cos(u_3) - mgcos(x_6)) \\ 2000 - x_1(end) \\ 50 - x_2(end) \\ x_1(1) \\ x_2(2) \\ 10 - x_4(1) \\ x_3(1) + 100 \end{bmatrix} = 0 \tag{2}$$

inequality constraints

$$\begin{bmatrix} x_4\sqrt{u_1^2 + u_2^2} - 415000 \\ \sqrt{u_1^2 + u_2^2} - 400 \end{bmatrix} \leq 0 \tag{3}$$

This is the graph is produced.
Next the normalized constraints were tested.

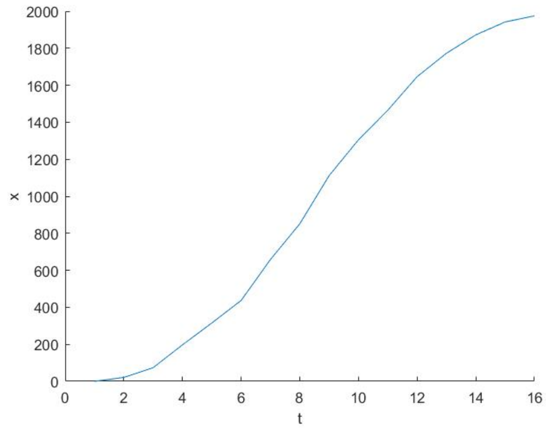
$$SCALE = quadrotor_model_parameters.scale_posn \tag{4}$$

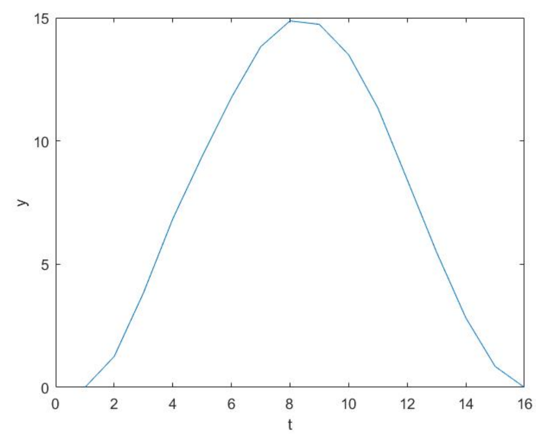


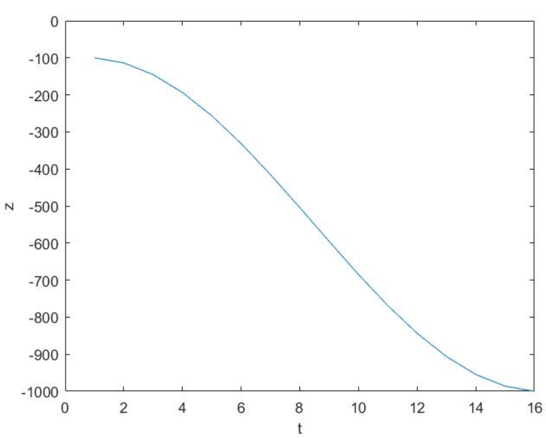
$$
\begin{bmatrix}
Da_1 - \frac{t_f}{2}(x_4cos(x_4)cos(x_45)) \\
Da_2 - \frac{t_f}{2}(x_4sin(x_4)cos(x_45)) \\
Da_3 - \frac{t_f}{2}(-x_4sin(x_5) \\
Da_4\frac{t_f}{2}(-\frac{1}{m}(u_1 - D - mgsin(x_6)/SCALE) \\
Da_5\frac{t_f}{2}\frac{u_2sin(u_3)}{mx_4cos(x_6)} \\
Da_6\frac{1}{mx_4}(u_2cos(u_3) - mgcos(x_6)/SCALE) \\
xif(1) - x_1(end) \\
xif(2) - x_2(end) \\
xi0(1) \\
xi0(2) \\
xi0(4) + x_4(1) \\
x_3(1) + xi0(3)
\end{bmatrix} = 0 \tag{5}
$$

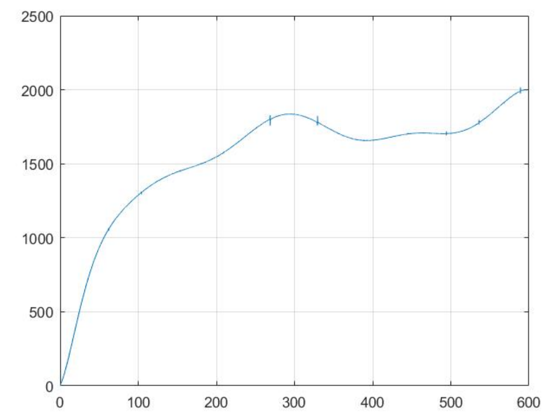
$$
\begin{bmatrix}
x_4\sqrt{u_1^2 + u_2^2} - 415000/SCALE^2 \\
\sqrt{u_1^2 + u_2^2} - 400/SCALE
\end{bmatrix} \leq 0 \tag{6}
$$

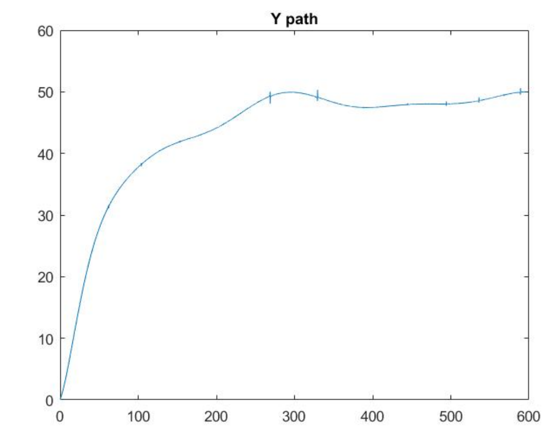
Here are the graphs produced.
This are the uncorrected graphs
The graphs produced with the the Legendre polynomials

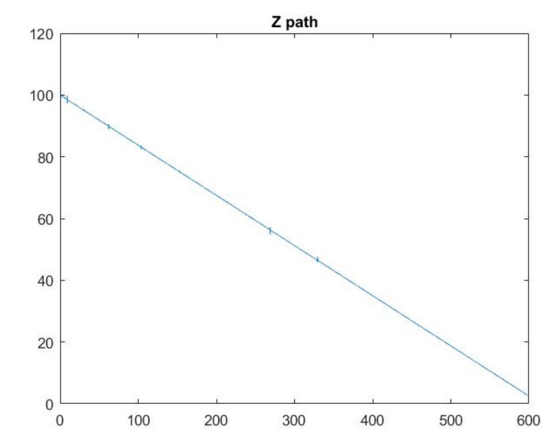

x
t

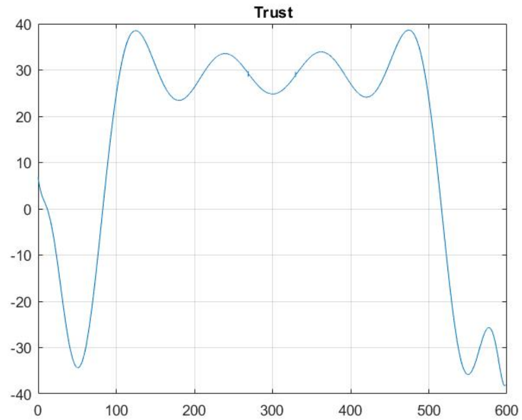

y
t

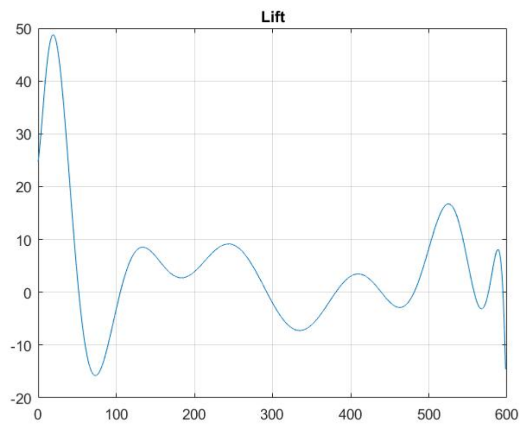

z
t

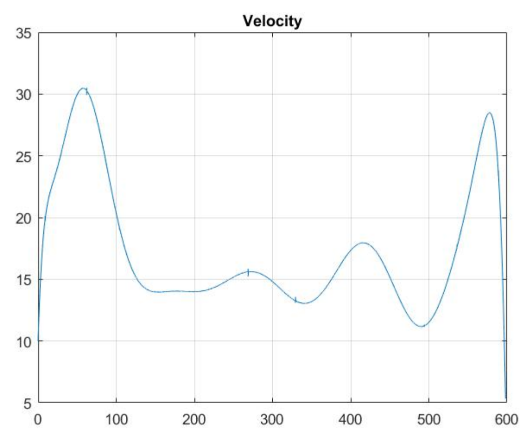

2500
2000
1500
1000
500
0
0
100
200
300
400
500
600


Y path
60
50
40
30
20
10
0
0
100
200
300
400
500
600


Z path
120
100
80
60
40
20
0
0
100
200
300
400
500
600


Trust
40
30
20
10
0
-10
-20
-30
-40
0
100
200
300
400
500
600

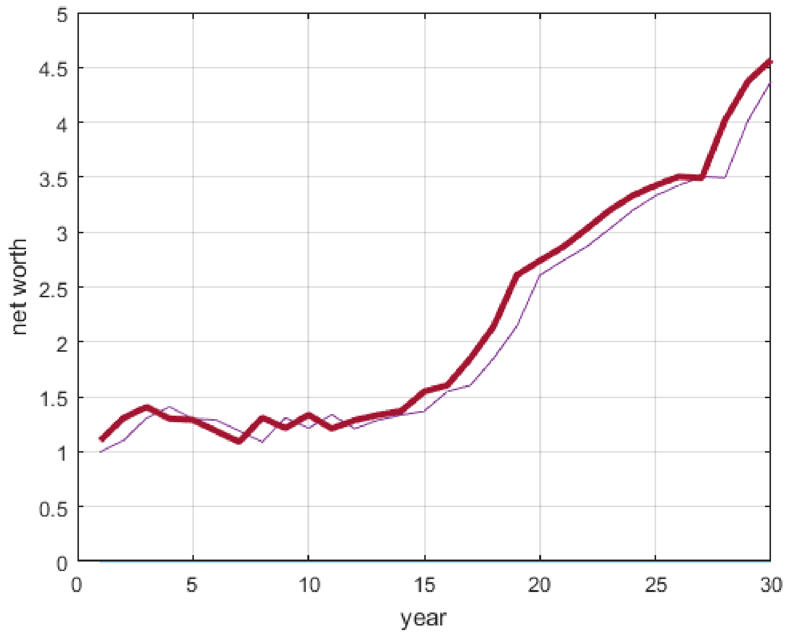

Lift
50
40
30
20
10
0
-10
-20
0
100
200
300
400
500
600


Velocity
35
30
25
20
15
10
5
0
100
200
300
400
500
600

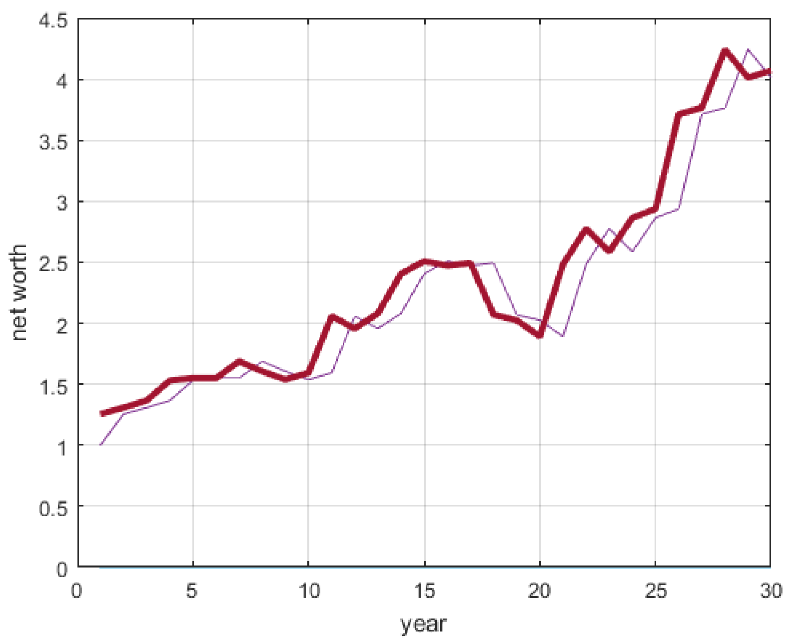
Problem 2

Problem 2

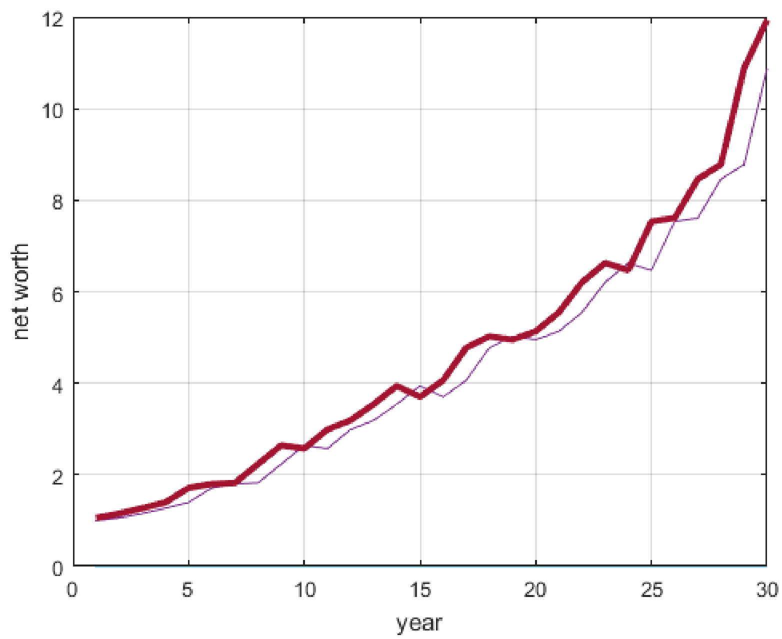
In this problem, we choose the risk-neutral approach to avoid any unnecessary lost during the investment. Also, we use the myopic policy method, which means we only care about the net worth for one year. The strategy for this approach is considering the asset that has highest expected value (i.e. the mean return) with reasonable standard deviation. Therefore, we come down to the 3rd and the 4th asset.
Then, we iteratively run the $portfolio\_sim\_main.m$ to get enough samples to find average final net worth.
If we choose to go all-in on the 4th asset, in the final year, the average final net worth is \$4.33 M. The maximum possible final net worth for this policy can go up to \$39.5 M. However, the minimum can go as low as \$0.475 M. These numbers come from running the main MATLAB code with this policy for 10,000 iterations
If we choose to equally invest in asset 3 and 4, the maximum possible net worth is around \$15 M and the minimum net worth is around \$1.3 M. The average net worth is also about \$4.3 M. The plots below show some of the net worth trend throughout 30 years if we decide to invest with myopic risk-neutral policy. Since we run the code for 100 iterations, the range is not very large but the average final net worth is still about \$4.33 M.



net worth
year



net worth
year



net worth
year
0
2
4
6
8
10
12
0
5
10
15
20
25
30

Problem 3

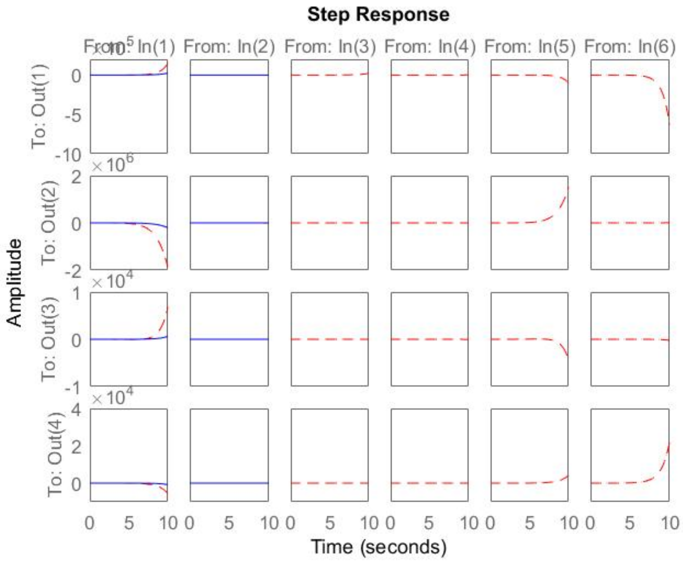
Problem 3

A kalman filter was combind with a LQR to from a LQG. They were compared sided by side. The graphs are hard to read but the LQG should do a better job of tracking then the Kalman filter

```
clear all; close all; clc
load learjet24_lin.mat

t_step = 0.1;
A = expm(A_long * t_step);
B = ( A_long \ (A - eye(4)) )*B_long ;
C = eye(4);
D = [0];
Q = 0.01*eye(4);
R = 0.1*eye(2);

V = diag([5 1 0.1 0.1])
W = diag([5 1 0.1 0.1])
G = eye(4);
H = [0];

sys = ss(A,B,C,D)
sysKalman = ss(A,[B G],C,[])

[K,S,E] = dlqr(A,B,Q,R);

sys2 = kalman(sysKalman ,V,W,[]);

F = lqgreg(sysKalman,K);

clsys = feedback(sys,F,+1);
step(sysKalman,'r--',clsys,'b-',10)
```

```
V =

    5.0000         0         0         0
         0    1.0000         0         0
         0         0    0.1000         0
         0         0         0    0.1000


W =

    5.0000         0         0         0
         0    1.0000         0         0
         0         0    0.1000         0
         0         0         0    0.1000


sys =

  A =
                x1          x2          x3          x4
   x1        0.998    0.001249     -0.1139      -3.217
   x2      -0.0192      0.9031       60.32     0.03315
   x3    0.0001033  -0.0009437       0.844  -0.0001647
   x4    5.114e-06  -4.909e-05     0.09256           1

  B =
                u1          u2
   x1      0.05285    -0.02295
   x2       -48.36     -0.1087
   x3       -1.325   -0.003216
   x4     -0.06816  -0.0001653

  C =
       x1  x2  x3  x4
   y1   1   0   0   0
   y2   0   1   0   0
   y3   0   0   1   0
   y4   0   0   0   1

  D =
       u1  u2
   y1   0   0
```

```
  y2   0   0
  y3   0   0
  y4   0   0

Continuous-time state-space model.


sysKalman =

  A =
                x1          x2          x3          x4
   x1        0.998    0.001249     -0.1139      -3.217
   x2      -0.0192      0.9031       60.32     0.03315
   x3    0.0001033  -0.0009437       0.844  -0.0001647
   x4    5.114e-06  -4.909e-05     0.09256           1

  B =
                u1          u2          u3          u4          u5          u6
   x1      0.05285    -0.02295           1           0           0           0
   x2       -48.36     -0.1087           0           1           0           0
   x3       -1.325   -0.003216           0           0           1           0
   x4     -0.06816  -0.0001653           0           0           0           1

  C =
       x1  x2  x3  x4
   y1   1   0   0   0
   y2   0   1   0   0
   y3   0   0   1   0
   y4   0   0   0   1

  D =
       u1  u2  u3  u4  u5  u6
   y1   0   0   0   0   0   0
   y2   0   0   0   0   0   0
   y3   0   0   0   0   0   0
   y4   0   0   0   0   0   0

Continuous-time state-space model.
```


Step Response
From: In(1)
From: In(2)
From: In(3)
From: In(4)
From: In(5)
From: In(6)
Amplitude
To: Out(1)
To: Out(2)
To: Out(3)
To: Out(4)
×10^5
×10^6
×10^4
×10^4
Time (seconds)