24-677 Project 1

P2

Qishun Yu

qishuny@andrew.cmu.edu

Given:

$$\frac{d}{dt}\begin{bmatrix} e_1 \\ \dot{e}_1 \\ e_2 \\ \dot{e}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{4C_\alpha}{mV_x} & \frac{4C_\alpha}{m} & -\frac{2C_\alpha(l_f-l_r)}{mV_x} \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{2C_\alpha(l_f-l_r)}{I_zV_x} & \frac{2C_\alpha(l_f-l_r)}{I_z} & -\frac{2C_\alpha(l_f^2+l_r^2)}{I_zV_x} \end{bmatrix}\begin{bmatrix} e_1 \\ \dot{e}_1 \\ e_2 \\ \dot{e}_2 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \frac{2C_\alpha}{m} & 0 \\ 0 & 0 \\ \frac{2C_\alpha l_f}{I_z} & 0 \end{bmatrix}\begin{bmatrix} \delta \\ F \end{bmatrix}$$

plug in
$m = 1888.6$
$C_\alpha = 20000$
$l_f = 1.55$
$l_r = 1.39$
$I_z = 25854$

$$\begin{bmatrix} \dot{e}_1 \\ \ddot{e}_1 \\ \dot{e}_2 \\ \ddot{e}_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-42.3594}{V_x} & 42.3594 & \frac{-3.3888}{V_x} \\ 0 & 0 & 0 & 1 \\ 0 & \frac{-0.2475}{V_x} & 0.2475 & \frac{-6.7063}{V_x} \end{bmatrix}}_{A} + \underbrace{\begin{bmatrix} 0 & 0 \\ 21.1797 & 0 \\ 0 & 0 \\ 2.3981 & 0 \end{bmatrix}}_{B}$$

where:
$V_x = 2, 5, 8$

Controllability

$$P = [B \quad AB \quad A^2B \quad A^3B]$$

Observability

$$Q = \begin{bmatrix} C \\ CA \\ CA^2 \\ CA^3 \end{bmatrix}$$

Using Matlab to test the controllability for all 3 values, the rank of P, Q are 4.
Therefore, the system is $\boxed{\text{Controllable \& observable}}$

2. a

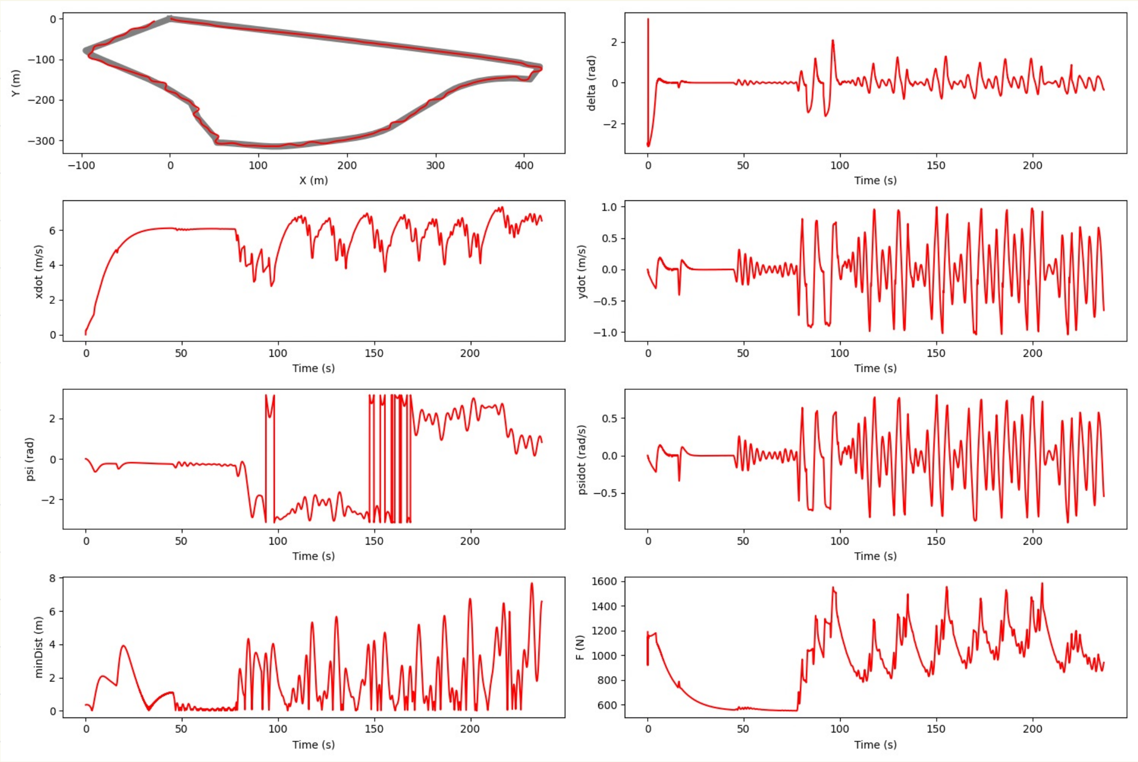
the system is more controllable because the σ_n is increasing. Thus the system is more likely to be full rank.

2. b

As a conclusion, the real parts of the poles are moving towards 0 as the speed increase. Therefore, the system is more unstable as the speed goes up.

Check Q1-pdf for plots

Performance



In [50]:
```
import numpy as np
import matplotlib.pyplot as plt
import control
from numpy.linalg import matrix_rank
```

Exercise 1. 1.

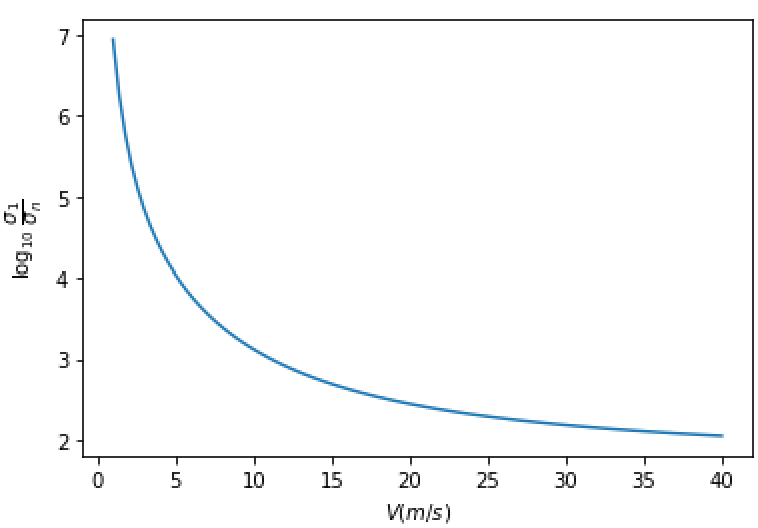
In [51]:
```
m = 1888.6;
ca = 20000;
lf = 1.55;
lr = 1.39;
iz = 25854;
for i in range(3):
    if i == 0:
        vx = 2;
    elif i == 1:
        vx = 5;
    else:
        vx = 8;
    A = np.array([[0,1,0,0],
        [0,-4*ca/(m*vx),4*ca/m,-2*ca*(lf-lr)/(m*vx)],
        [0,0,0,1],
        [0,-2*ca*(lf-lr)/(iz*vx),2*ca*(lf-lr)/iz,-2*ca*(lf**2+lr**2)/(iz*vx
)]]);
    B = np.array([[0,0],
        [2*ca/m,0],
        [0,0],
        [2*ca*lf/iz,0]]);

    C = np.array([[1,0,0,0],
        [0,1,0,0],
        [0,0,1,0],
        [0,0,0,1]]);

    P = np.hstack((B, np.matmul(A, B), np.matmul(np.linalg.matrix_power(A, 2),
B), np.matmul(np.linalg.matrix_power(A, 3), B)))
    rP = matrix_rank(P)
    print('the rank of the controllability matrix when vx = ', vx, 'is')
    print(rP)
    Q = np.vstack((C, np.matmul(C, A), np.matmul(C, np.linalg.matrix_power(A,
2)), np.matmul(C, np.linalg.matrix_power(A, 3))))
    rQ = matrix_rank(Q)
    print('the rank of the observability matrix when vx = ', vx, 'is')
    print(rQ)
print('therefore the system is controllable and observable')
```

```
the rank of the controllability matrix when vx =  2 is
4
the rank of the observability matrix when vx =  2 is
4
the rank of the controllability matrix when vx =  5 is
4
the rank of the observability matrix when vx =  5 is
4
the rank of the controllability matrix when vx =  8 is
4
the rank of the observability matrix when vx =  8 is
4
therefore the system is controllable and observable
```

Exercise 1. 2.a

In [52]:
```
v = np.linspace(1, 40, 100)

y = []
real = []
for i in range(4):
    real.append([])
for i in range(len(v)):
    vTemp = v[i]
    A = np.array([[0,1,0,0],
        [0,-4*ca/(m*vTemp),4*ca/m,-2*ca*(lf-lr)/(m*vTemp)],
        [0,0,0,1],
        [0,-2*ca*(lf-lr)/(iz*vTemp),2*ca*(lf-lr)/iz,-2*ca*(lf**2+lr**2)/(iz*vT
emp)]]);
    B = np.array([[0,0],
        [2*ca/m,0],
        [0,0],
        [2*ca*lf/iz,0]]);
    C = np.array([[1,0,0,0],
        [0,1,0,0],
        [0,0,1,0],
        [0,0,0,1]]);
    D = np.array([[0,0],[0,0],[0,0],[0,0]])
    P = np.hstack((B, np.matmul(A, B), np.matmul(np.linalg.matrix_power(A, 2),
B), np.matmul(np.linalg.matrix_power(A, 3), B)))
    _, s, _ = np.linalg.svd(P)
    s1 = max(s)
    sn = min(s)
    y.append(np.log10(s1 / sn))
    sys = control.StateSpace(A, B, C, D)
    poles = control.pole(sys)
    for j in range(4):
        real[j].append(poles[j].real)
y = np.array(y)
real = np.array(real)
plt.figure()
plt.plot(v,y)
plt.xlabel('$V (m/s)$')
plt.ylabel('$\log_{10}$ $\dfrac{\sigma_1}{\sigma_n}$')
```

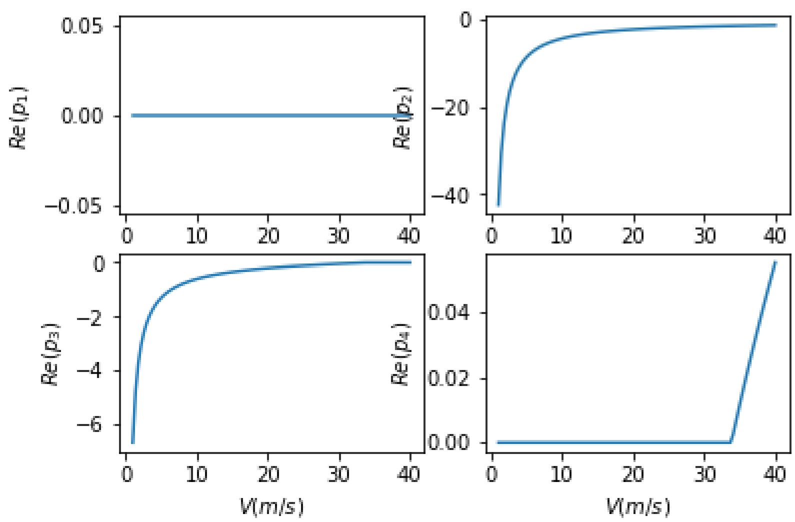
Out[52]:
```
Text(0, 0.5, '$\\log_{10}$ $\\dfrac{\\sigma_1}{\\sigma_n}$')
```



Exercise 1. 2.b

In [53]:
```
plt.figure()

plt.subplot(2, 2, 1)
plt.plot(v, real[0])
plt.xlabel('$V (m/s)$')
plt.ylabel('$Re(p_1)$')

plt.subplot(2, 2, 2)
plt.plot(v, real[1])
plt.xlabel('$V (m/s)$')
plt.ylabel('$Re(p_2)$')

plt.subplot(2, 2, 3)
plt.plot(v, real[2])
plt.xlabel('$V (m/s)$')
plt.ylabel('$Re(p_3)$')

plt.subplot(2, 2, 4)
plt.plot(v, real[3])
plt.xlabel('$V (m/s)$')
plt.ylabel('$Re(p_4)$')
```

Out[53]:
```
Text(0, 0.5, '$Re(p_4)$')
```



In [ ]:

In [ ]: