Homework 1

Name: Shaobo Wang

Exercise 1. Types of Systems

1.1.

$\forall\alpha,\beta\neq 0, \alpha y_u(u_1)+\beta y_u(u_2)=0=y_u(\alpha u_1+\beta u_2)$

Assume $u_2(t)=u_1(t-\tau)$, $y_2(t)=y_u[u_2(t)]=y_u[u_1(t-\tau)]=0=y_1(t-\tau)$

This system is **linear** and **time invariant**;

1.2.

$\forall\alpha,\beta\neq 0, \alpha y_1+\beta y_2=\alpha u_1^3+\beta u_2^3\neq y_u(\alpha u_1+\beta u_2)=(\alpha u_1+\beta u_2)^3$

Assume $u_2(t)=u_1(t-\tau)$, $y_2(t)=u_2^3(t)=u_1^3(t-\tau)=y_1(t-\tau)$

This system is **non-linear** and **time invariant**;

1.3.

$\forall\alpha,\beta\neq 0, \alpha y_1(t)+\beta y_2(t)=\alpha u_1(3t)+\beta u_2(3t)=y_u(\alpha u_1(t)+\beta u_2(t))$

Assume $u_2(t)=u_1(t-\tau)$, $y_2(t)=u_2(3t)=u_1[3(t-\tau)]=y_1(t-\tau)$

This system is **linear** and **time invariant**;

1.4.

$\forall\alpha,\beta\neq 0,$
$\alpha y_1(t)+\beta y_2(t)=\alpha e^{-t}u_1(t-T)+\beta e^{-t}u_2(t-T)=e^{-t}[\alpha u_1(t-T)+\beta u_2(t-T)]=y_u(\alpha u_1(t)+\beta u_2(t))$

Assume $u_2(t)=u_1(t-\tau)$,
$y_2(t)=e^{-t}u_2(t-T)=e^{-t}u_1[(t-T)-\tau]\neq e^{-(t-\tau)}u_1[(t-\tau)-T]=y_1(t-\tau)$

This system is **linear** and **time variant**;

1.5.

If $t>0$, $\forall\alpha,\beta\neq 0, \alpha y_1(t)+\beta y_2(t)=\alpha u_1(t)+\beta u_2(t)=y_u(\alpha u_1(t)+\beta u_2(t))$

If $t\leq 0$, $\forall\alpha,\beta\neq 0, \alpha y_1+\beta y_2=0=y_u(\alpha u_1+\beta u_2)$

Assume $u_2(t)=u_1(t-\tau),\tau>0,$

If $0<\tau<t$, $y_2(t)=u_2(t)=u_1(t-\tau)=y_1(t-\tau)$

If $0<t<\tau$, $y_2(t)=u_2(t)=u_1(t-\tau)\neq y_1(t-\tau)=0$

If $t<0<\tau$, $y_2(t)=0=y_1(t-\tau)$

This system is **linear** and **time variant.**

Exercise 2. State space representations

2.1.

$$p_i(k+1) = p_i(k) \cdot \alpha\gamma \cdot \frac{q_i(k)}{s_i(k)} = p_i(k) \cdot \alpha\gamma \cdot \frac{\sigma^2 + \sum_{i \neq j} G_{ij} p_j(k)}{G_{ii} p_i(k)} = \sum_{i \neq j} \alpha\gamma \frac{G_{ij}}{G_{ii}} p_j(k) + \frac{\alpha\gamma}{G_{ii}} \sigma^2$$

$$p_1(k+1) = \alpha\gamma \frac{G_{12}}{G_{11}} p_2(k) + \alpha\gamma \frac{G_{13}}{G_{11}} p_3(k) + \frac{\alpha\gamma}{G_{11}} \sigma^2$$

$$p_2(k+1) = \alpha\gamma \frac{G_{21}}{G_{22}} p_1(k) + \alpha\gamma \frac{G_{23}}{G_{22}} p_3(k) + \frac{\alpha\gamma}{G_{22}} \sigma^2$$

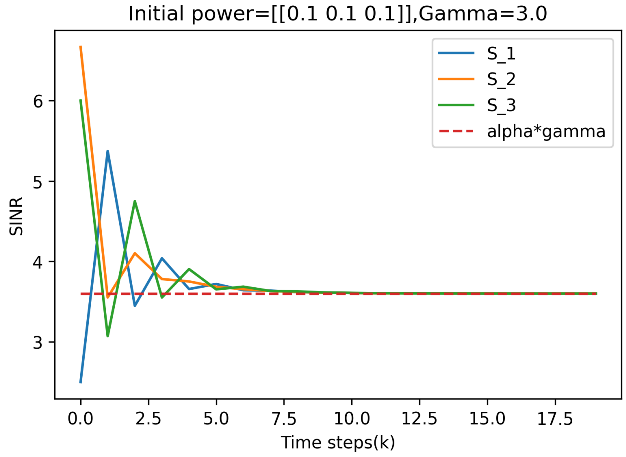
$$p_3(k+1) = \alpha\gamma \frac{G_{31}}{G_{33}} p_1(k) + \alpha\gamma \frac{G_{32}}{G_{33}} p_2(k) + \frac{\alpha\gamma}{G_{33}} \sigma^2$$

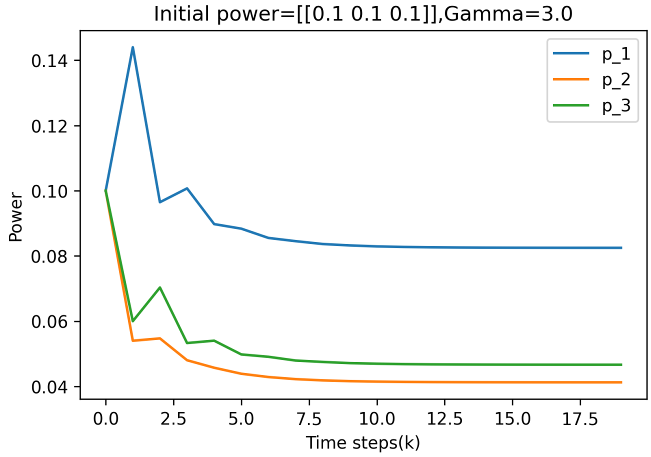
Assume $A = \begin{bmatrix} 0 & \alpha\gamma \frac{G_{12}}{G_{11}} & \alpha\gamma \frac{G_{13}}{G_{11}} \\ \alpha\gamma \frac{G_{21}}{G_{22}} & 0 & \alpha\gamma \frac{G_{23}}{G_{22}} \\ \alpha\gamma \frac{G_{31}}{G_{33}} & \alpha\gamma \frac{G_{32}}{G_{33}} & 0 \end{bmatrix}$, $B = [\frac{\alpha\gamma}{G_{11}}, \frac{\alpha\gamma}{G_{22}}, \frac{\alpha\gamma}{G_{33}}]^T$,

The algorithm can be expressed as $p(k+1) = Ap(k) + B\sigma^2$.

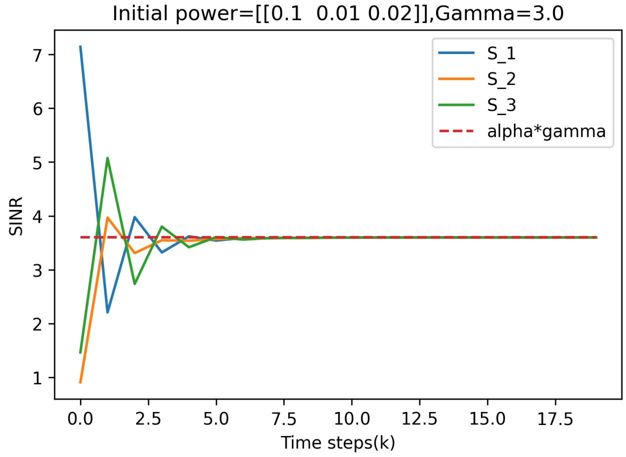
2.2.

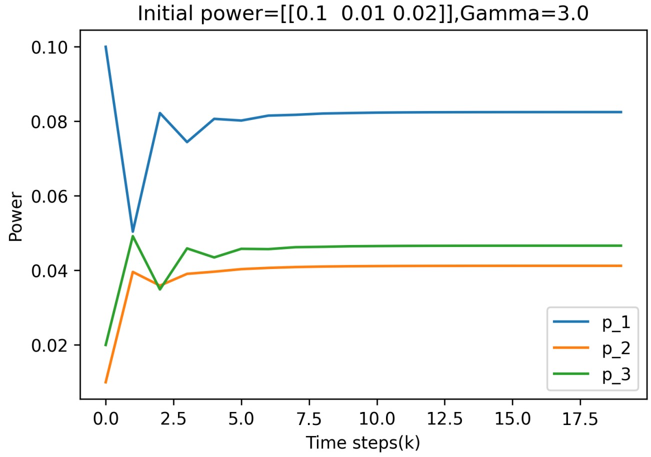
For initial conditions: $p_1 = p_2 = p_3 = 0.1$, and $\gamma = 3$:





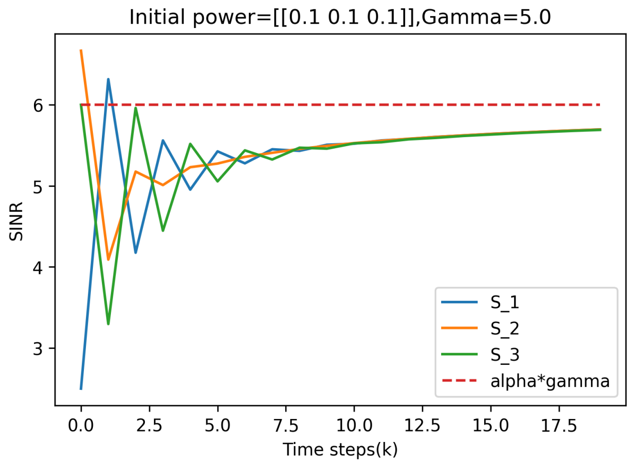
For initial conditions: $p_1 = 0.1, p_2 = 0.01, p_3 = 0.02$, and $\gamma = 3$:

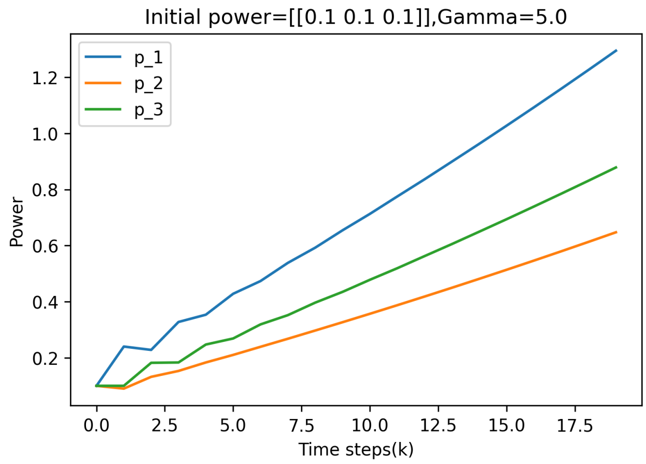




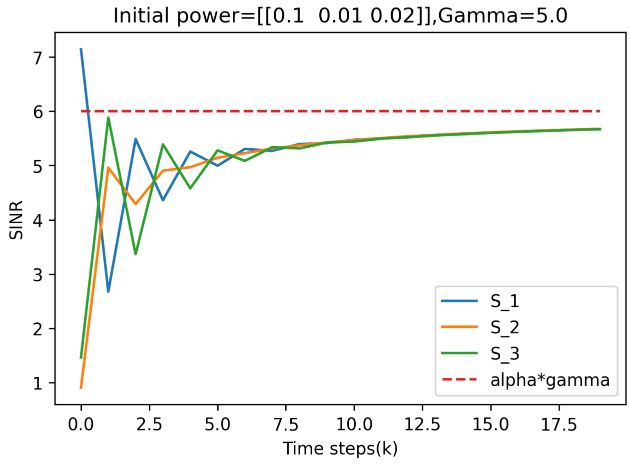
As shown in the plot, S_i is able to achieve the goal under the situation above.

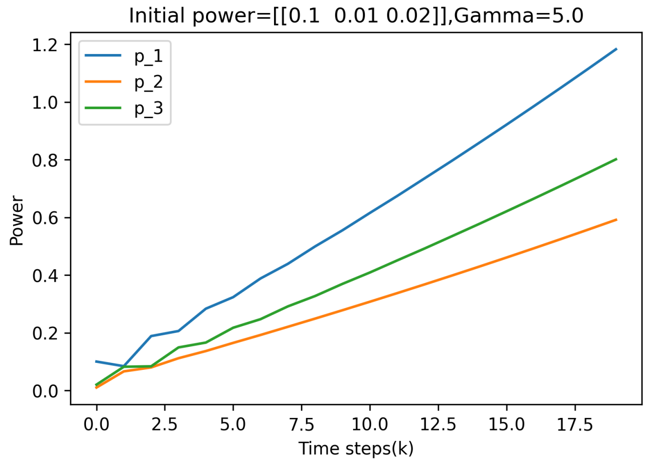
For initial conditions: $p_1 = p_2 = p_3 = 0.1$, and $\gamma = 5$:





For initial conditions: $p_1 = 0.1, p_2 = 0.01, p_3 = 0.02$, and $\gamma = 5$:





As shown in the plot, S_i is not able to achieve the goal under the situation above, and the power level diffuses.

Exercise 3. Linearization

Assume $x_1 = y, x_2 = \dot{y}$, $\dot{x}_1 = \dot{y}, \dot{x}_2 = \ddot{y}$,

$$\dot{x} = [x_2, -\frac{x_1^3}{2} + 2x_1 - (1 + x_1)x_2]^T$$

$$\begin{cases} x_2 = 0 \\ -\frac{x_1^3}{2} + 2x_1 - x_2 - x_1 x_2 = 0 \end{cases} \Rightarrow x_2 = 0; x_1 = \pm 2 \; or \; x_1 = 0$$

The given differential equation has 3 equilibrium points: $(0,0)$, $(2,0)$, $(-2,0)$

The Jacobian matrix $\dfrac{\partial f}{\partial x} = \begin{bmatrix} 0 & 1 \\ -\frac{3}{2}x_1^2 - x_2 + 2 & -x_1 - 1 \end{bmatrix}$

Around $(0,0)$: $\dot{\delta}_x = \begin{bmatrix} 0 & 1 \\ 2 & -1 \end{bmatrix} \delta_x$,

The linearized equation is: $\ddot{y} + \dot{y} - 2y = 0$

Around $(2,0)$: $\dot{\delta}_x = \begin{bmatrix} 0 & 1 \\ -4 & -3 \end{bmatrix} \delta_x$,

The linearized equation is: $\ddot{y} + 3\dot{y} + 4y - 8 = 0$

Around $(-2,0)$: $\dot{\delta}_x = \begin{bmatrix} 0 & 1 \\ -4 & 1 \end{bmatrix} \delta_x$,

The linearized equation is: $\ddot{y} - \dot{y} + 4y + 8 = 0$

Exercise 4. Equilibrium

To find the equilibrium points, let $\dot{x} = 0$,

$$\begin{cases} x_2 = 0 \\ -g(\frac{D}{x_1+D})^2 + \frac{\ln(u)}{m} = 0 \end{cases} \Rightarrow x_2^* = 0; x_1^* = \pm D\sqrt{\frac{mg}{\ln(u)}} - D$$

The equilibrium states $(x_1^*, x_2^*) = (D\sqrt{\frac{mg}{\ln(u)}} - D, 0) or (-D\sqrt{\frac{mg}{\ln(u)}} - D, 0)$

The Jacobian Matrix $\dfrac{\partial f}{\partial x} = \begin{bmatrix} 0 & 1 \\ \frac{2gD^2}{(x_1+D)^3} & 0 \end{bmatrix}$

Around $(D\sqrt{\frac{mg}{\ln(u)}} - D, 0)$, $\dot{\delta}_x = \begin{bmatrix} 0 & 1 \\ \frac{2g}{D[\frac{mg}{\ln(u)}]^{\frac{3}{2}}} & 0 \end{bmatrix} \delta_x$,

The linearized model is: $\ddot{x}_1 = \dfrac{2g}{D[\frac{mg}{\ln(u)}]^{\frac{3}{2}}} x_1 - \dfrac{2g(\sqrt{\frac{mg}{\ln(u)}} - 1)}{[\frac{mg}{\ln(u)}]^{\frac{3}{2}}}$

Around $(-D\sqrt{\frac{mg}{\ln(u)}} - D, 0)$, $\dot{\delta}_x = \begin{bmatrix} 0 & 1 \\ -\frac{2g}{D[\frac{mg}{\ln(u)}]^{\frac{3}{2}}} & 0 \end{bmatrix} \delta_x$,

The linearized model is: $\ddot{x}_1 = -\dfrac{2g}{D[\frac{mg}{\ln(u)}]^{\frac{3}{2}}} x_1 - \dfrac{2g(\sqrt{\frac{mg}{\ln(u)}} - 1)}{[\frac{mg}{\ln(u)}]^{\frac{3}{2}}}$

Since x_1 is supposed to be positive, only $(D\sqrt{\frac{mg}{\ln(u)}} - D, 0)$ is feasible.

Hence, **The final linearized model should be:** $\ddot{x}_1 = \dfrac{2g}{D[\frac{mg}{\ln(u)}]^{\frac{3}{2}}} x_1 - \dfrac{2g(\sqrt{\frac{mg}{\ln(u)}} - 1)}{[\frac{mg}{\ln(u)}]^{\frac{3}{2}}}$

Exercise 5. Linearization

5.1.

The satellite is on the reference orbit, $r(t) \equiv p$ and $\theta(t) = \omega t$.

Hence,

$\ddot{r} = \dot{r} = 0$ and $\dot{\theta} = \omega, \ddot{\theta} = 0$

The normalized equations of motion follow:

$0 = p\omega^2 - \frac{k}{p^2}$, i.e. $k = p^3\omega^2$

5.2.

Assume $x_1 = r$, $x_2 = \dot{r}$, $x_3 = \theta$, $x_4 = \dot{\theta}$,

The state space follows:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} x_2 \\ x_1 x_4^2 - \frac{k}{x_1^2} + u_1 \\ x_4 \\ -2\frac{x_4}{x_1} x_2 + \frac{u_2}{x_1} \end{bmatrix}$$

The configuration point $x = [x_1, x_2, x_3, x_4]^T$ is constrained by the orbit.

Hence,

$$x^* = [x_1^*, x_2^*, x_3^*, x_4^*]^T = [p, 0, \omega t, \omega]^T$$

The Jacobian Matrix of the state space equation around x^* is:

$$\frac{\partial f}{\partial x}|_{x*} = \lim_{x \to x^*} \begin{bmatrix} 0 & 1 & 0 & 0 \\ x_4^2 + \frac{2k}{x_1^3} & 0 & 0 & 2x_1 x_4 \\ 0 & 0 & 0 & 1 \\ \frac{2x_2 x_4 - u_2}{x_1^2} & -\frac{2x_4}{x_1} & 0 & -\frac{2x_2}{x_1} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \omega^2 + \frac{2k}{p^3} & 0 & 0 & 2p\omega \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{2\omega}{p} & 0 & 0 \end{bmatrix}$$

Since $k = p^3 \omega^2$, $\frac{\partial f}{\partial x}|_{x*} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 3\omega^2 & 0 & 0 & 2p\omega \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{2\omega}{p} & 0 & 0 \end{bmatrix}$

The linearized equation about this orbit can be described in state space:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 3\omega^2 & 0 & 0 & 2p\omega \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{2\omega}{p} & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 - p \\ x_2 \\ x_3 - \omega t \\ x_4 - \omega \end{bmatrix}$$

Code for Exercise 2.2 simulation and plotting:

```
import numpy as np
import matplotlib.pyplot as plt

def sim2Mat(power0,gamma):
  ## constant
  gain=np.array([[1,.2,.1],[.1,2,.1],[.3,.1,3]])
  alpha=1.2
  sigma=0.1

  ## control rules
  A_mat=np.zeros([3,3])
  for i in range(3):
    for j in range(3):
        if i==j:
          A_mat[i,j]=0.
        else:
          A_mat[i,j]=alpha*gamma*gain[i,j]/gain[i,i]
```

```
  B_mat=np.zeros([3,1])
  for i in range(3):
    B_mat[i]=alpha*gamma/gain[i,i]

  ## simulate
  power=power0
  P_mat=np.empty([3,0]) # power through time
  S_mat=np.empty([3,0]) # sinr through time

  k=np.arange(0,20) # time steps
  for n in k:

 sinr=np.array([gain[0,0]*power[0]/(sigma**2+gain[0,1]*power[1]+gain[0,2]*power[2]),gain[1,
1]*power[1]/(sigma**2+gain[1,0]*power[0]+gain[1,2]*power[2]),gain[2,2]*power[2]/(sigma**2+g
ain[2,0]*power[0]+gain[2,1]*power[1])])
    S_mat=np.append(S_mat,sinr,axis=1)
    P_mat=np.append(P_mat,power,axis=1)
    power=A_mat@power+B_mat*(sigma**2)

  return [S_mat,P_mat]

power0_1=np.array([[.1],[.1],[.1]]) # initial power
power0_2=np.array([[.1],[.01],[.02]])
gamma_1=3. # threshold
gamma_2=5.
alpha=1.2
index=1

k=np.arange(0,20) # time steps

for gamma in [gamma_1,gamma_2]:
  for power in [power0_1,power0_2]:
    [S_mat,P_mat]=sim2Mat(power,gamma)

    plt.figure(index,dpi=300)
    plt.plot(k,S_mat[0,:],label='S_1')
    plt.plot(k,S_mat[1,:],label='S_2')
    plt.plot(k,S_mat[2,:],label='S_3')
    plt.plot(k,[alpha*gamma for k in range(0,20)],linestyle='dashed',label='alpha*gamma')
    plt.legend()
    plt.xlabel("Time steps(k)")
    plt.ylabel("SINR")
    plt.title("Initial power="+str(power.T)+",Gamma="+str(gamma))

    plt.figure(index+1,dpi=300)
    plt.plot(k,P_mat[0,:],label='p_1')
    plt.plot(k,P_mat[1,:],label='p_2')
    plt.plot(k,P_mat[2,:],label='p_3')
    plt.legend()
    plt.xlabel("Time steps(k)")
    plt.ylabel("Power")
    plt.title("Initial power="+str(power.T)+",Gamma="+str(gamma))

    plt.show()
    index=index+2
```