OptoElec & ONIP-1 / TD Systèmes et Signaux

Séance 1 / Systèmes asservis

Exercice 1 / Modélisation numérique d'un système / ALI

Soit un système $A(j\omega)$ de type passe-bas, d'ordre 1, de gain statique A_0 et de pulsation de coupure w_0.

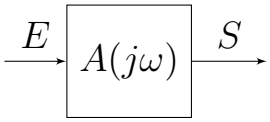
1. Donner la fonction de transfert de ce système.
2. A l'aide de la bibliothèque **control** sous Python, définir la fonction de transfert de ce système à l'aide de la fonction *tf*.
3. Tracer la **réponse en fréquence** de ce système à l'aide de la fonction *bode_plot*.
4. Tracer la **réponse indicielle** à l'aide de la fonction *step_response*, puis **impulsionnelle** à l'aide de la fonction *impulse_response*.
5. Calculer et afficher la **transformée de Fourier** de la réponse impulsionnelle. Que pouvez-vous en conclure ?

Applications Numériques : $A_0 = 10^5$, $f_0 = 30\,\text{Hz}$.

Fonction de transfert :

$$A(j\omega) = \frac{A_0}{1 + j \cdot \frac{\omega}{\omega_0}}$$

Représentation sous forme d'un schéma bloc :

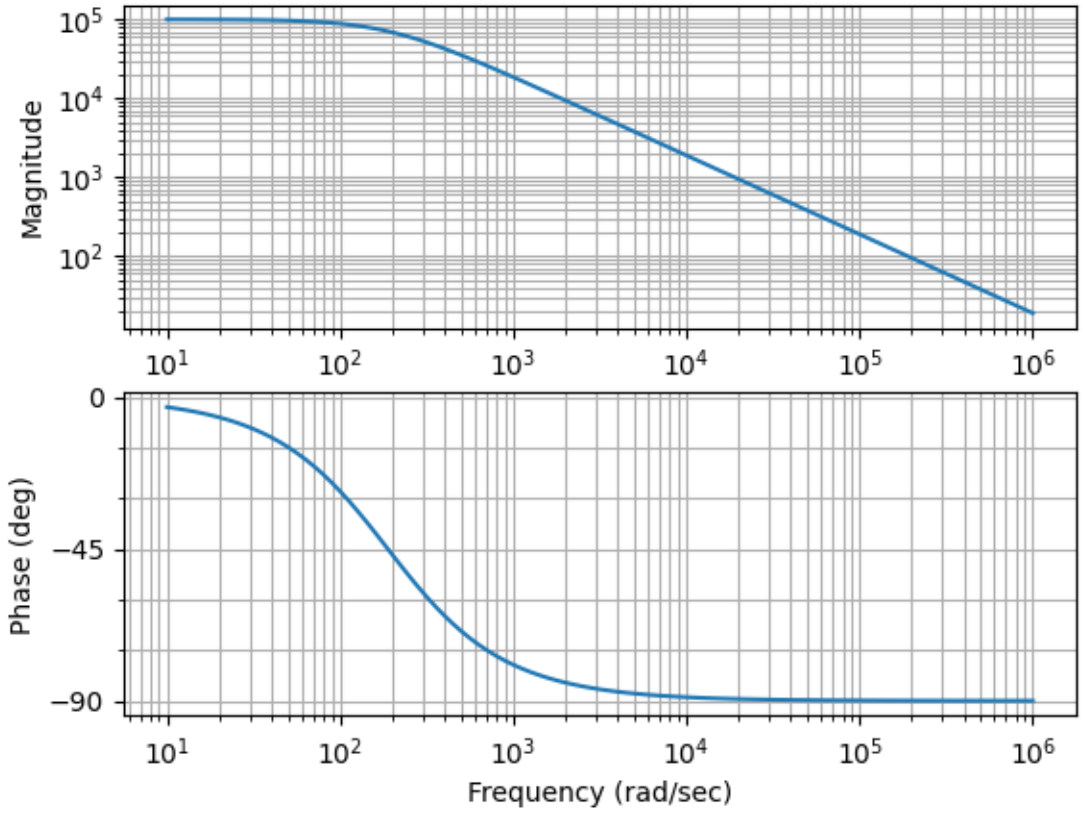


Définition du système avec la bibliothèque **control** :

```
1  import control as ct
2
3  A0 = 1e5
4  f0 = 30
5  w0 = 2 * np.pi * f0
6
7  num_A = [A0]
8  den_A = [1/w0, 1]
9  sys_A = ct.tf(num_A, den_A)
```

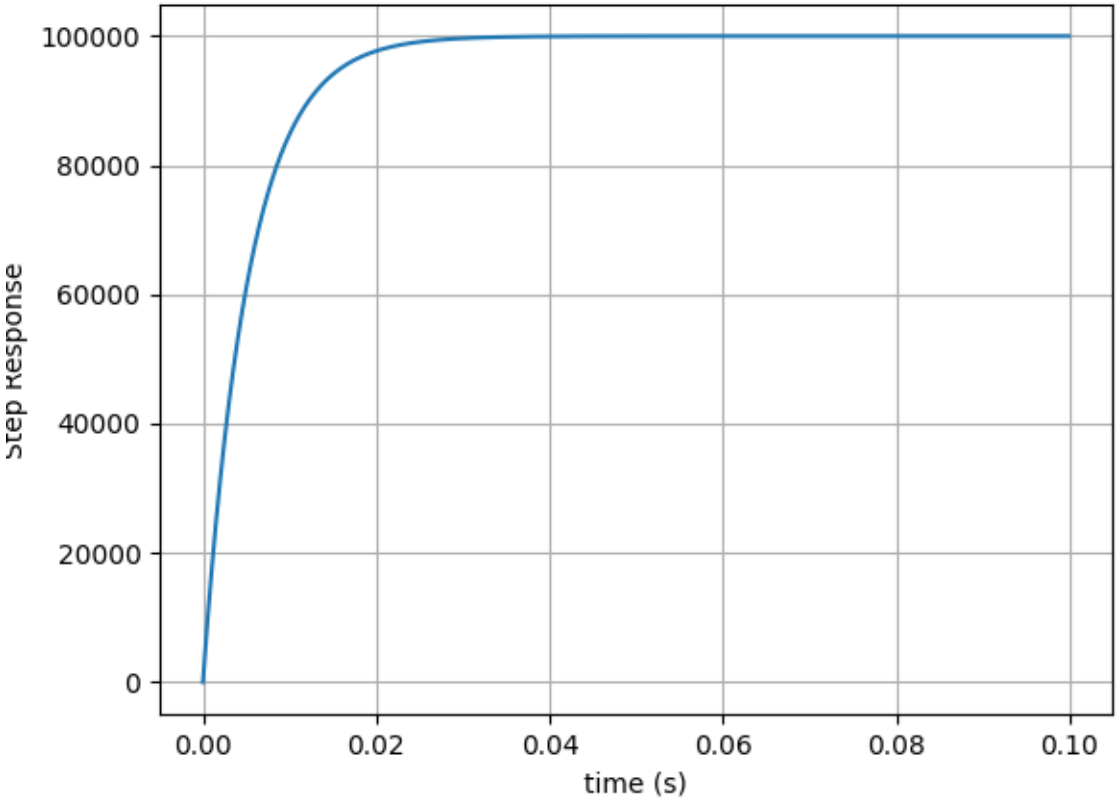
Diagramme de Bode du système :

```
1  # Bode diagram
2  w = np.logspace(1, 6, 101)
3  ct.bode_plot([sys_A], w, name='A')
4  plt.show()
```



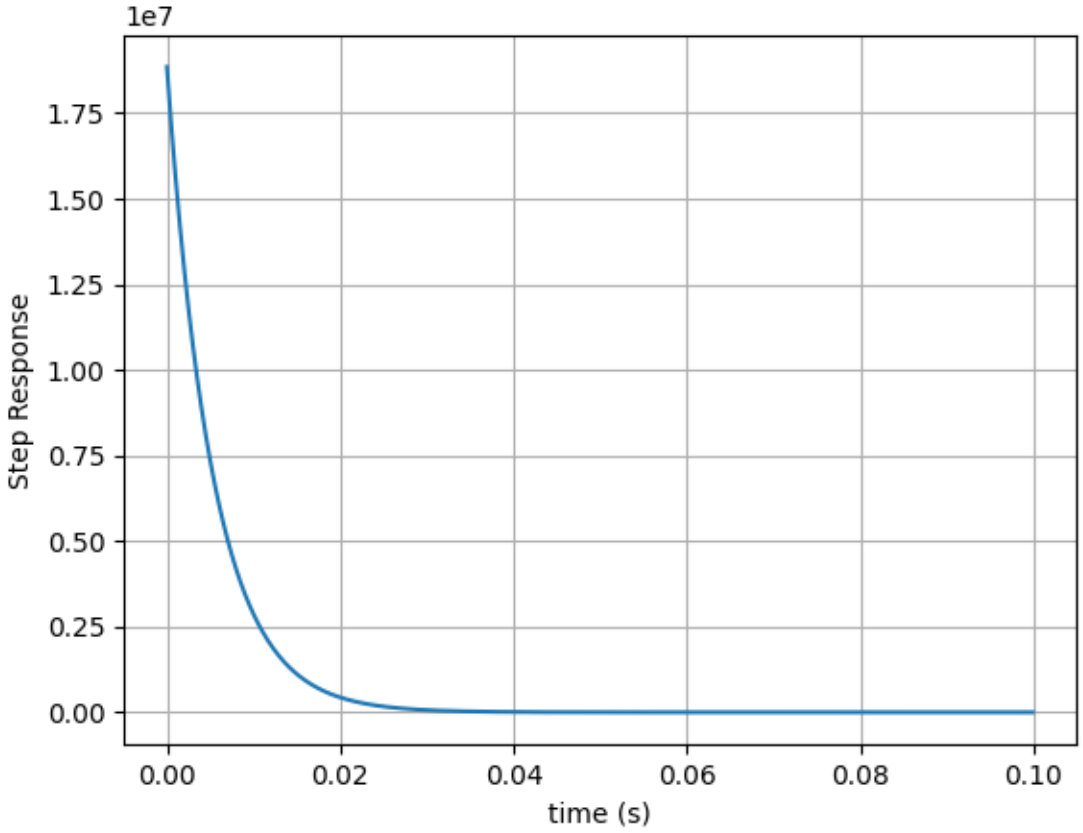
Réponse indicielle du système :

```
time = np.arange(0, 0.1, 0.0001)

T, yout = ct.step_response(sys_A, time)
plt.figure()
plt.plot(T, yout)
plt.xlabel("time (s)")
plt.ylabel("Step Response")
plt.grid()
```



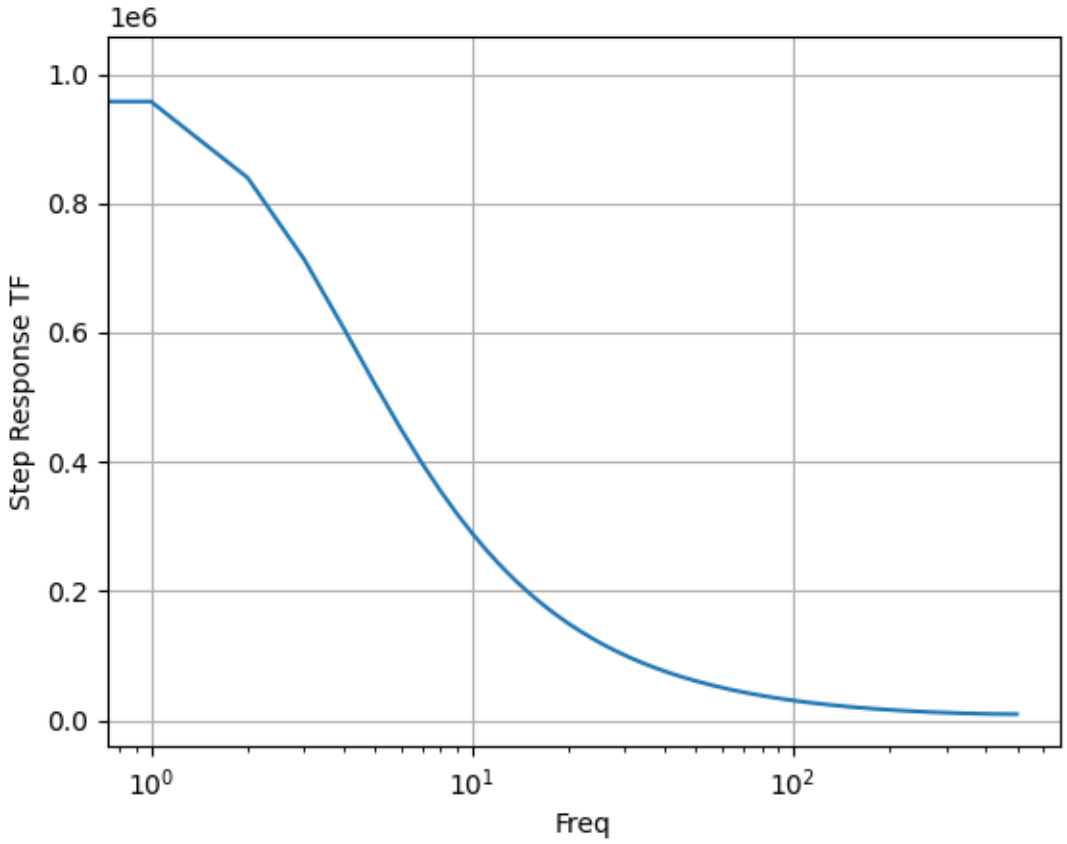
Réponse impulsionnelle du système :

```
1  T, yout = ct.impulse_response(sys_A, time)
2  plt.figure()
3  plt.plot(T, yout)
4  plt.xlabel("time (s)")
5  plt.ylabel("Step Response")
6  plt.grid()
```



Transformée de Fourier discrète de la réponse impulsionnelle du système :

```
tf_yout = np.fft.fft(yout)
tf_yout = tf_yout / len(tf_yout)
plt.figure()
plt.semilogx(np.abs(tf_yout[:len(tf_yout)//2]))
plt.xlabel("Freq")
plt.ylabel("Step Response TF")
plt.grid()
plt.show()
```



Exercice 2 / Rebouclage d'un système - Système asservi / ALI en régime "linéaire"

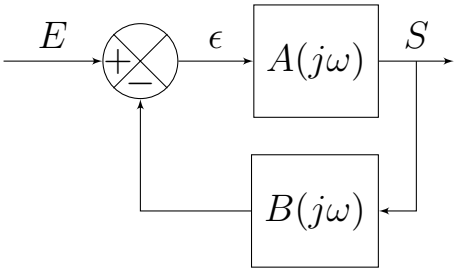
Il est possible de reboucler un système à l'aide d'un autre système. On parle alors d'un système asservi.

On prendra ici le système A pour la boucle d'action et un système $B(j\omega) = 1/K$, où K est une constante, comme système de contre-réaction.

1. Tracer le **schéma bloc** de ce système puis donner la **fonction de transfert** de ce système.
2. Que vaut le **produit** du gain statique par la fréquence de coupure de ce système ? Comparer cette valeur à celui du système A. Que pouvez-vous en conclure ?
3. Définir la **fonction de transfert** du système B à l'aide de la fonction *tf*.
4. Définir un système C qui est le système complet avec la rétro-action, à l'aide de la fonction *feedback*.
5. Tracer la **réponse en fréquence** des systèmes A et C à l'aide de la fonction *bode_plot* sur un même graphique.
6. Tracer la **réponse indicielle** des systèmes A et C à l'aide de la fonction *step_response* sur un même graphique.

Applications Numériques : $A_0 = 10^5$, $f_0 = 30\,\text{Hz}$ et $K = 1$ puis $K = 10$.

Schéma bloc du système :



Fonction de transfert du système :

On peut calculer

$$\epsilon = E - B(j\omega) \cdot S$$

$$S = A(j\omega) \cdot \epsilon$$

En regroupant ces expressions on obtient :

$$\frac{S}{E} = \frac{A(j\omega)}{1 + A(j\omega) \cdot B(j\omega)}$$

En remplaçant $A(j\omega)$ et $B(j\omega)$ par leurs expressions respectives, on obtient :

$$H_{BF} = \frac{S}{E} = \frac{A_0}{\frac{A_0}{K} + 1} \cdot \frac{1}{1 + \frac{j\omega}{\omega_0 \cdot (1 + \frac{A_0}{K})}}$$

On peut alors identifier le gain statique et la fréquence de coupure de ce nouveau système :

$H_{BF0} = \frac{A_0}{\frac{A_0}{K}+1}$ et $\omega_{BF0} = \omega_0 \cdot (1 + \frac{A_0}{K})$

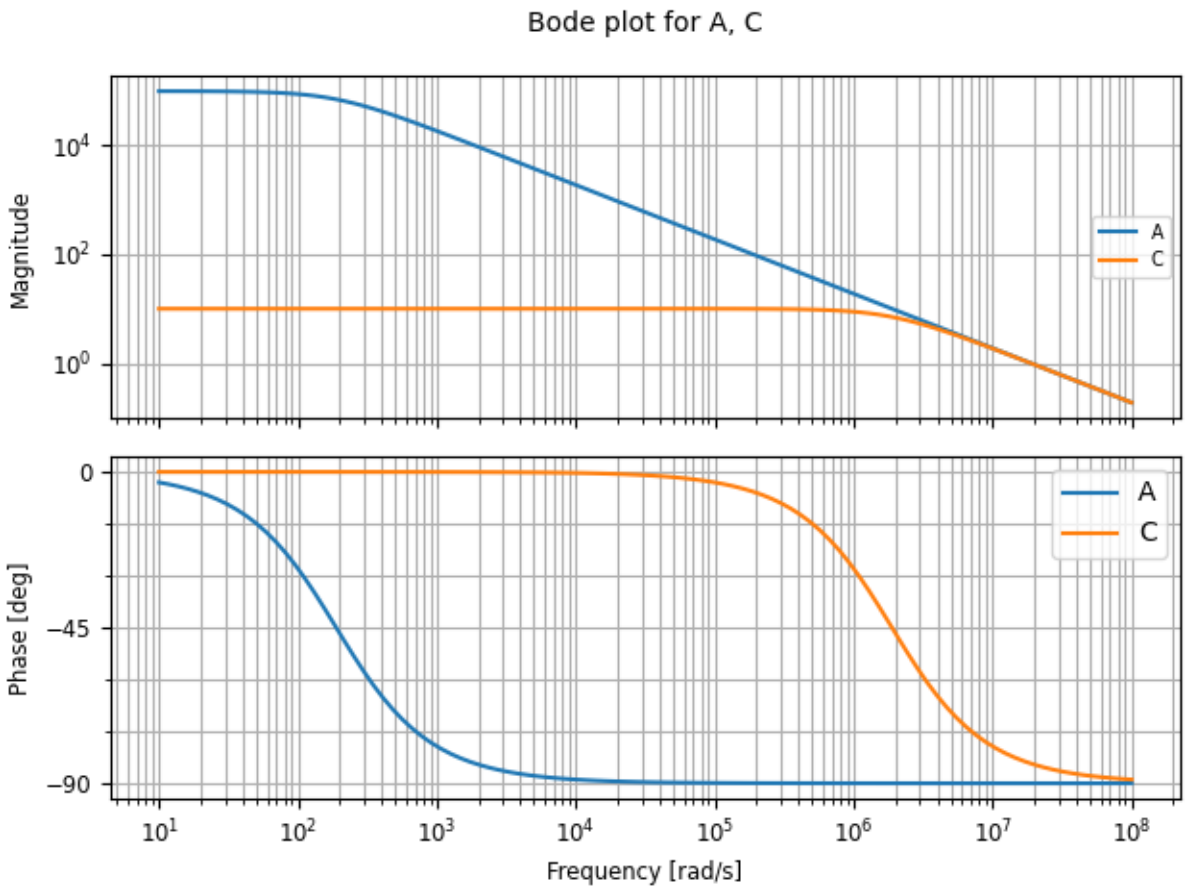
Le produit des deux donne : $H_{BF0} \cdot \omega_{BF0} = A_0 \cdot \omega_0 = cte$

Définition des systèmes B et C :

```
K = 10
sys_B = ct.tf([1],[K], name='B')

sys_C = ct.feedback(sys_A, sys_B, name='C')
```

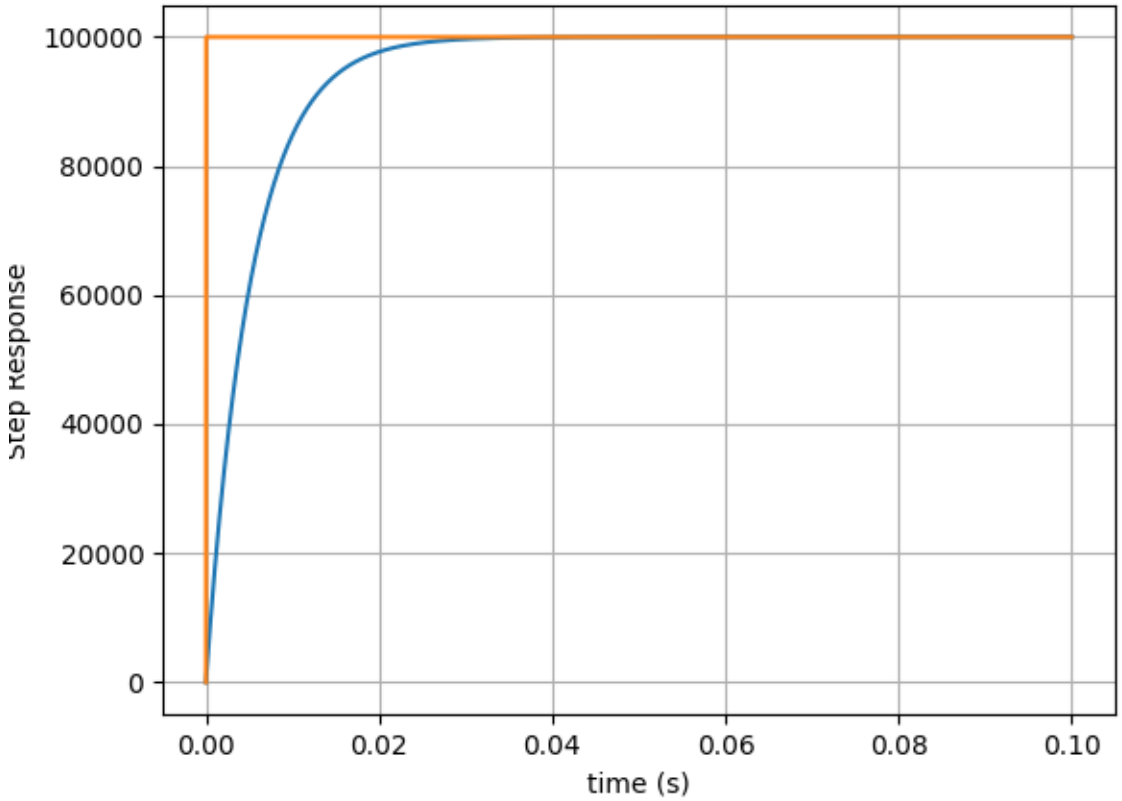
Diagramme de Bode des systèmes A et C :

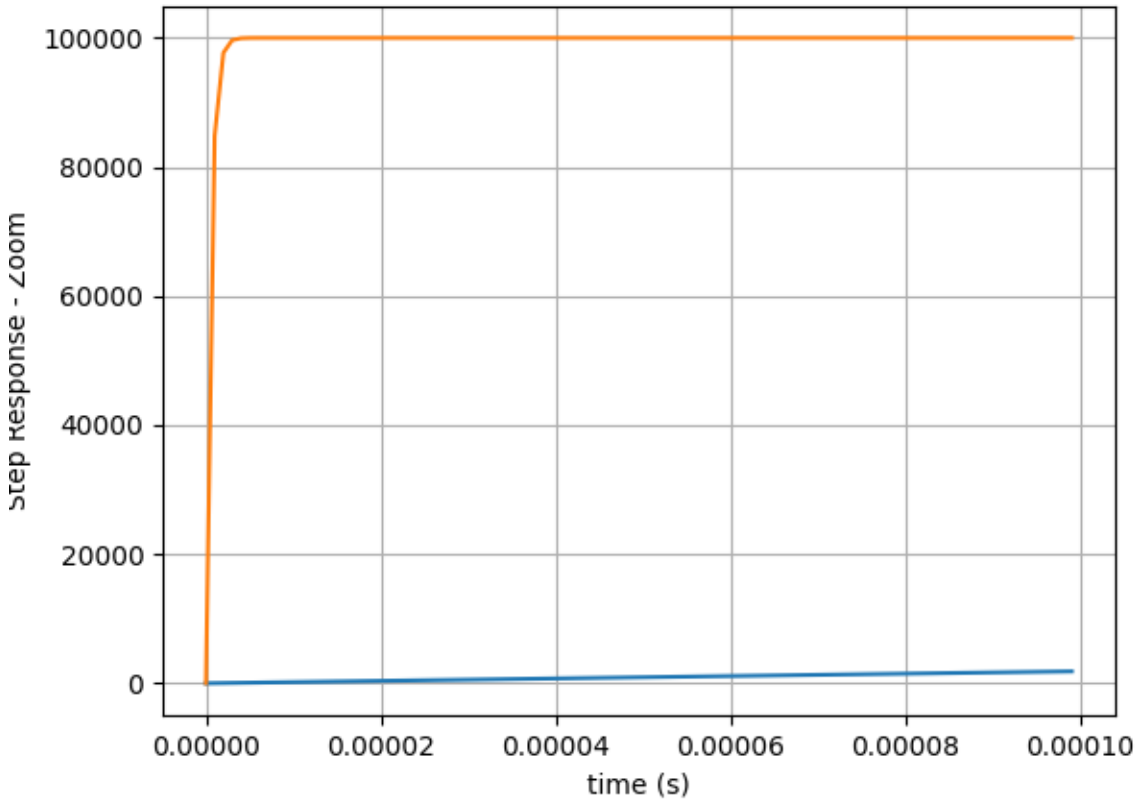
```
w = np.logspace(1, 8, 101)
ct.bode_plot([sys_A, sys_C], w)
plt.show()
```



Réponse indicielle des systèmes A et C :

```
time = np.arange(0, 0.1, 0.000001)

T, yout_A = ct.step_response(sys_A, time)
T, yout_C = ct.step_response(sys_C, time)
plt.figure()
plt.plot(T, yout_A, label='A')
plt.plot(T, 1e4*yout_C, label='C (*1e4)')
plt.xlabel("time (s)")
plt.ylabel("Step Response")
plt.grid()
plt.show()
```


Step Response
100000
80000
60000
40000
20000
0
0.00
0.02
0.04
0.06
0.08
0.10
time (s)


Step Response - Zoom
100000
80000
60000
40000
20000
0
0.00000
0.00002
0.00004
0.00006
0.00008
0.00010
time (s)

Exercice 3 / Système du second ordre

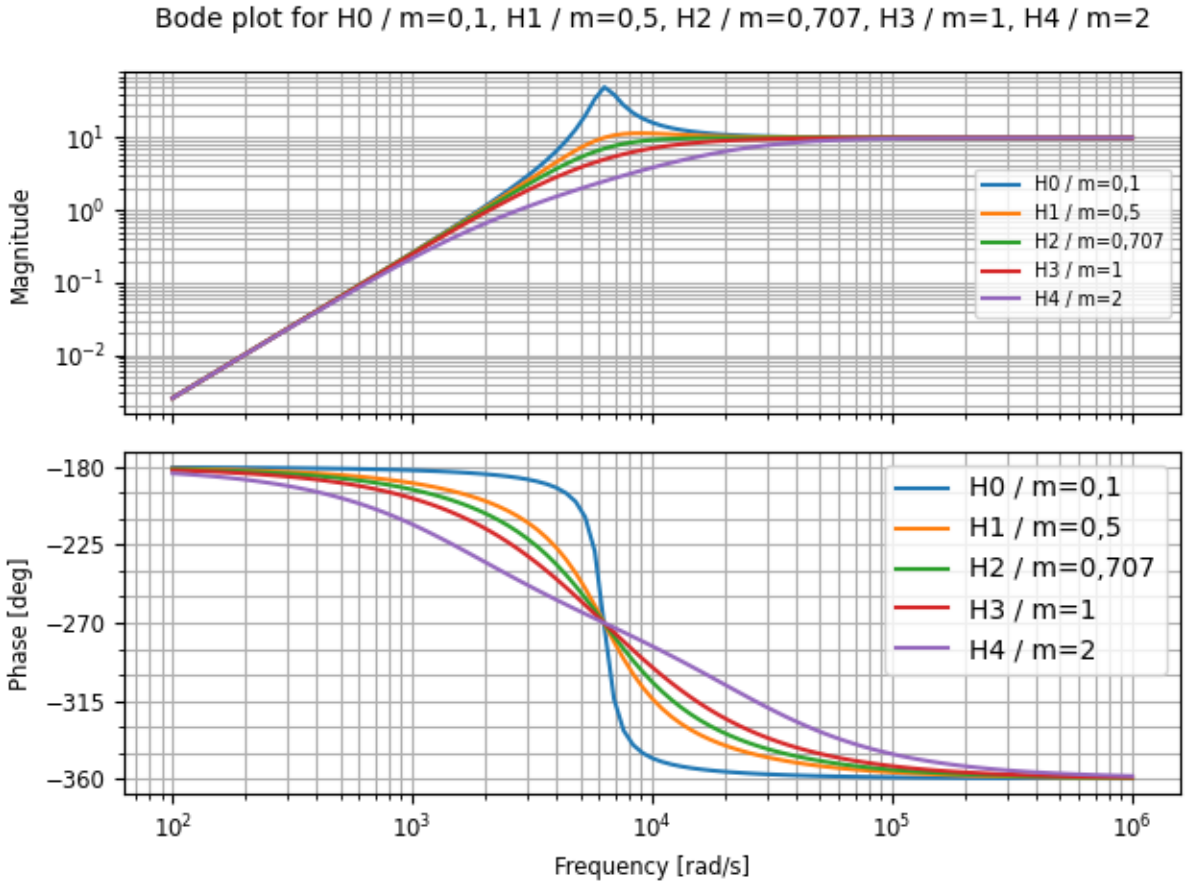
On se propose de simuler un système du second ordre dont la fonction de transfert peut être mise sous la forme suivante :

$$H(j\omega) = \frac{A_0 \cdot (\frac{j \cdot \omega}{\omega_0})^2}{1 + 2 \cdot m \cdot \frac{j \cdot \omega}{\omega_0} + (\frac{j \cdot \omega}{\omega_0})^2}$$

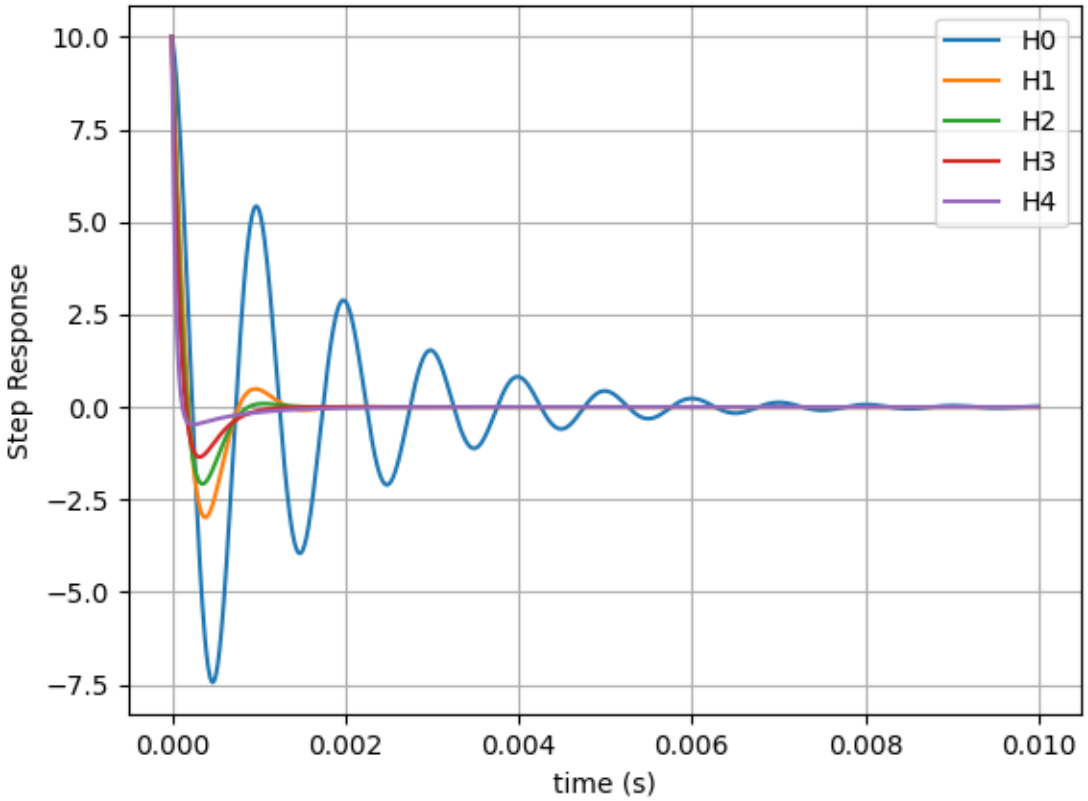
1. Définir ce système.
2. Tracer la réponse en fréquence de ce système pour $m = [0.1, 0.5, 0.7, 1.0, 2]$ sur un même graphique.
3. Tracer la réponse indicielle de ce système pour $m = [0.1, 0.5, 0.7, 1.0, 2]$ sur un même graphique.

Applications Numériques : $A_0 = 10$, $f_0 = 1000\,\text{Hz}$.

```python
import control as ct
import numpy as np
from matplotlib import pyplot as plt

# Parameters of the system
A0 = 10     # V/V
f0 = 1000         # Hz
w0 = 2*np.pi*f0    # rd/s
m = [0.1, 0.5, 0.707, 1, 2]
m_n = ['0,1', '0,5', '0,707', '1', '2']

sys_H = []

for i, mm in enumerate(m):
    num = [A0/(w0**2), 0, 0]
    den = [1/w0**2, 2*float(mm)/w0, 1]
    nameH = f'H{i} / m={m_n[i]}'
    sys = ct.tf(num, den, name=nameH)
    sys_H.append(sys)

# Bode diagram
w = np.logspace(2, 6, 101)
ct.bode_plot(sys_H, w)
plt.legend()
```




```python
# Step Response
time = np.arange(0, 0.01, 0.00001)
plt.figure()
for i, sys in enumerate(sys_H):
    T, yout_H = ct.step_response(sys, time)
    plt.plot(T, yout_H, label=f'H{i}')
plt.xlabel("time (s)")
plt.ylabel("Step Response")
plt.grid()
plt.legend()
plt.show()
```


H0
H1
H2
H3
H4
Step Response
time (s)
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
0.000
0.002
0.004
0.006
0.008
0.010

Bibliothèque CONTROL

Plus d'aide sur la bibliothèque **control** : https://python-control.readthedocs.io/en/0.10.1/

Pour importer la bibliothèque **control** :

```
1 import control as ct
```

Définir un système

La **définition d'un système** par l'intermédiaire d'une fonction de transfert se fait à l'aide de la fonction *tf* :

```
1 # System with a transfert function : (3s + 4) / (6s^2 + 5s + 4).
2 num = [3, 4]
3 den = [6, 5, 4]
4 sys1 = ct.tf(num, den)
```

Tracer la réponse en fréquence d'un système

Il est possible de tracer la réponse en fréquence d'un système à l'aide de la fonction *bode_plot* :

```
1 ct.bode_plot(sys1)
2 plt.show()
```

Attention : cette fonction se base sur la bibliothèque Pyplot de Matplotlib. Il est indispensable de l'importer et de faire appel à la fonction show() pour visualiser les graphiques.

Il est également possible de passer une liste de système en argument de la fonction *bode_plot* afin de comparer plusieurs systèmes entre eux.

Tracer la réponse indicielle d'un système

Il est possible de tracer la réponse à un échelon d'un système à l'aide de la fonction *step_response* :

```
1 time = np.arange(0, 0.1, 0.0001)
2 T, yout = ct.step_response(sys1, time)
```

Cette fonction renvoie deux vecteurs : un vecteur temps (T) et le signal de sortie de la réponse à l'échelon (*yout*). Le vecteur *time* n'est pas indispensable. S'il n'est pas fourni, il est automatiquement calculé par la fonction *step_response*.

Pour pouvoir afficher le graphique associé, il est indispensable d'utiliser une bibliothèque graphique de type Pyplot de Matplotlib.

Tracer la réponse impulsionnelle d'un système

Il est possible de tracer la réponse à une impulsion d'un système à l'aide de la fonction *impulse_response* :

```
1 time = np.arange(0, 0.1, 0.0001)
2 T, yout = ct.impulse_response(sys1, time)
```

Cette fonction renvoie le même type de données que *step_response*

Reboucler un système

Il est possible de reboucler un système par un autre système à l'aide de la fonction *feedback* :

```
1 sys = ct.feedback(sys1, sys2)
```