

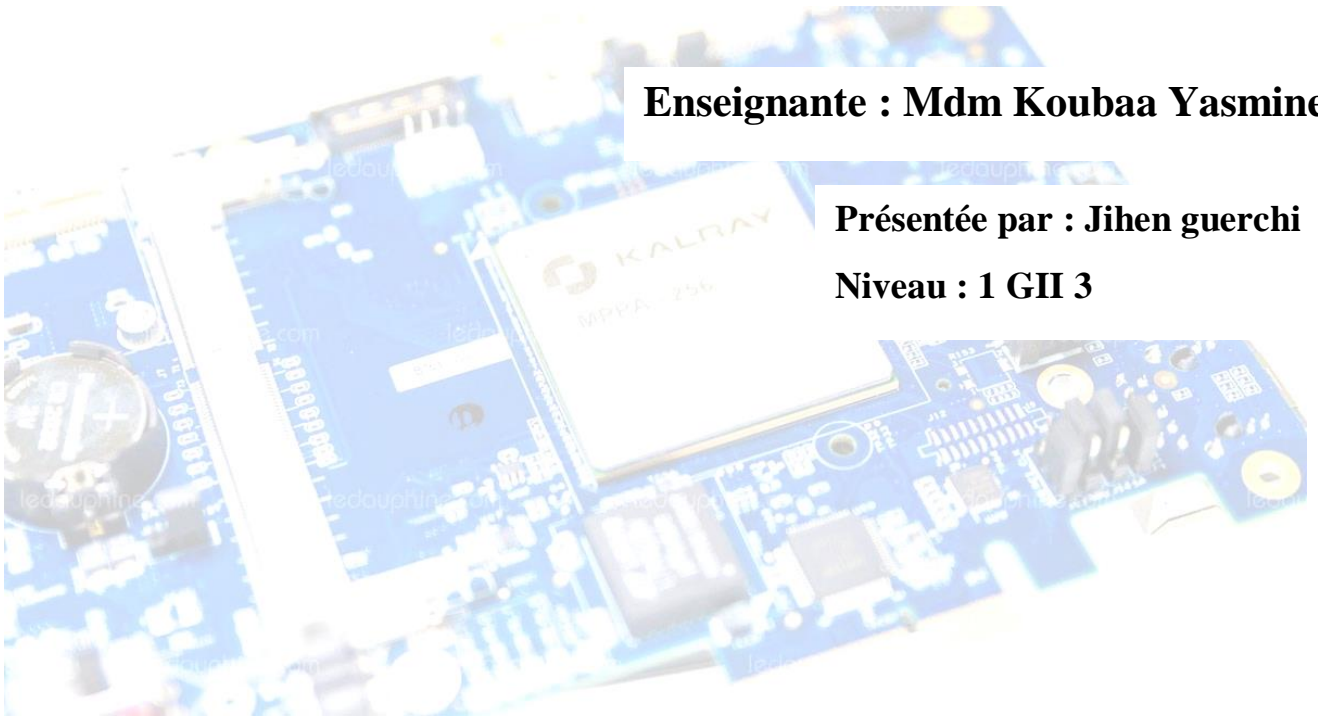
**2022/2023**

# **Fonction de transfert sous Python**

**Enseignante : Mdm Koubaa Yasmine**

**Présentée par : Jihen guerchi**

**Niveau : 1 GII 3**



## I. Introduction

Python est largement utilisé dans le domaine de l'automatisme pour automatiser les tâches, contrôler les processus, collecter et analyser des données, et créer des interfaces utilisateur graphiques. Les avantages de Python dans ce domaine comprennent sa simplicité, sa flexibilité, sa grande bibliothèque standard et sa compatibilité avec de nombreuses plates-formes et dispositifs d'automatisation.

## II. Les objectifs

- Connaitre les bibliothèques nécessaires sous python qui permettent de manipuler des fonctions de transfert.
- Apprendre l'écriture d'une fonction de transfert en utilisant le langage python.
- Apprendre à dessiner des figures des différentes réponses étudiées.

## III. Partie Réalisation

### 1) Préparation de l'environnement de travail

- Installation de l'anaconda

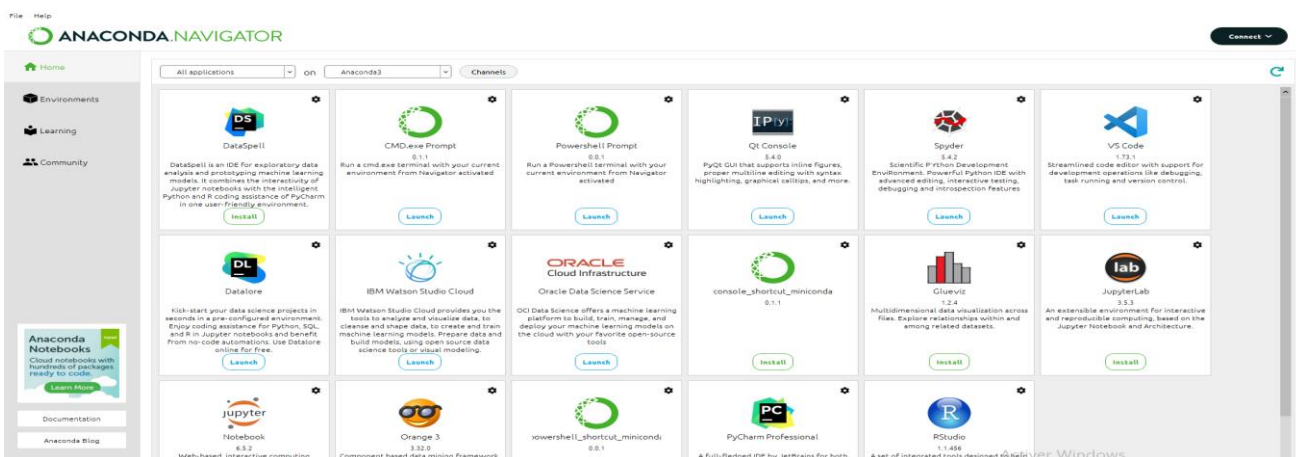


Figure : installation de l'anaconda

- Installation des bibliothèques nécessaires

## Mini\_Projet\_Python

A.U : 2022/2023

Nom & Prénom : Guerchi Jihen

Classe : 1<sup>ère</sup> GII

Groupe : 3

✓ Install de **control**

```
Administrateur : C:\Windows\ x + v

(base) C:\Users\Mesrs>pip install control
Collecting control
  Downloading control-0.9.3.post2-py3-none-any.whl (432 kB)
    432.8/432.8 kB 162.0 kB/s eta 0:00:00
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\site-packages (from control) (1.24.2)
Collecting matplotlib
  Downloading matplotlib-3.7.1-cp310-cp310-win_amd64.whl (7.6 MB)
    7.6/7.6 MB 271.1 kB/s eta 0:00:00
Collecting scipy>=1.3
  Downloading scipy-1.10.1-cp310-cp310-win_amd64.whl (42.5 MB)
    8.8/42.5 MB 49.5 kB/s eta 0:11:20
```

## Installation de **numpy**

```
(base) C:\Users\Mesrs>pip install numpy
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\site-packages (1.24.2)

(base) C:\Users\Mesrs>
```

## Installation de **Matplotlib**

```
(base) C:\Users\Mesrs>pip install matplotlib
Collecting matplotlib
  Using cached matplotlib-3.7.1-cp310-cp310-win_amd64.whl (7.6 MB)
Collecting cycler>=0.10
  Using cached cycler-0.11.0-py3-none-any.whl (6.4 kB)
Requirement already satisfied: packaging>=20.0 in c:\programdata\anaconda3\lib\site-packages (from matplotlib) (23.0)
Collecting contourpy>=1.0.1
  Downloading contourpy-1.0.7-cp310-cp310-win_amd64.whl (162 kB)
    143.4/163.0 kB 57.6 kB/s eta 0:00:01
```

➔ On peut travailler aussi avec le **Google Colab**

## Mini\_Projet\_Python

A.U : 2022/2023

Nom & Prénom : Guerchi Jihen

Classe : 1<sup>ère</sup> GII

Groupe : 3

### Importation des librairies

```
import numpy as np
import control as Co
from scipy import signal
import matplotlib.pyplot as plt
%matplotlib inline
```

✓ [55] pip install control

Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheels/>  
Requirement already satisfied: control in /usr/local/lib/python3.9/dist-packages (0.9.2)  
Requirement already satisfied: matplotlib in /usr/local/lib/python3.9/dist-packages (3.5.2)  
Requirement already satisfied: scipy>=1.3 in /usr/local/lib/python3.9/dist-packages (1.7.3)  
Requirement already satisfied: numpy in /usr/local/lib/python3.9/dist-packages (1.24.2)  
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.9/dist-packages (3.1.0)  
Requirement already satisfied: importlib-resources>=3.2.0 in /usr/local/lib/python3.9/dist-packages (5.12.0)  
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.9/dist-packages (1.0.7)  
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.9/dist-packages (1.4.5)  
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.9/dist-packages (9.5.0)  
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.9/dist-packages (2.8.2)  
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.9/dist-packages (23.1)  
Requirement already satisfied: cycycler>=0.10 in /usr/local/lib/python3.9/dist-packages (0.10.0)  
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.9/dist-packages (4.22.0)  
Requirement already satisfied: zipp>=3.1.0 in /usr/local/lib/python3.9/dist-packages (3.15.0)  
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.9/dist-packages (1.16.0)

## Quelques exemples d'apprentissage

**Exemple :**  $f(s) = (s^2 + 2s + 4) / ((s+3)(s+1))$


```
#exemple de fonction de transfert : f(p)= (s^2+2s+4)/((s+3)*(s+1))
num=[1,2,4]
den=np.convolve([1,3],[1,1])
sys1=Co.tf(num,den)
print(sys1)
```

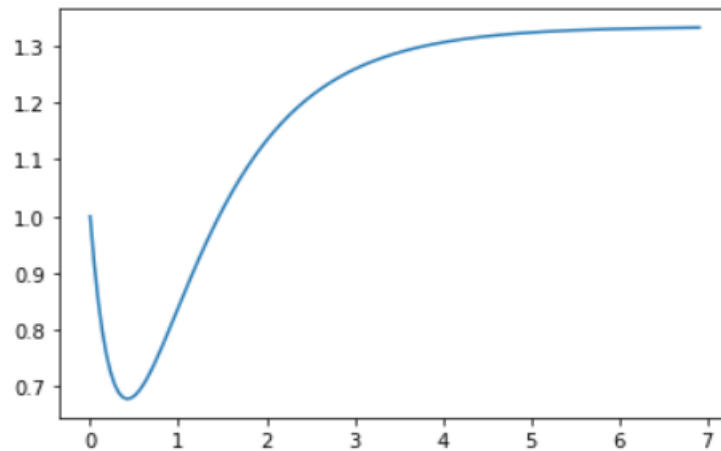
$$\frac{s^2 + 2s + 4}{s^2 + 4s + 3}$$

### Réponse indicielle

```
✓ [39] t,y=Co.step_response(sys1)
      plt.figure(1)
      plt.plot(t,y)
```

## Courbe de la réponse indicielle

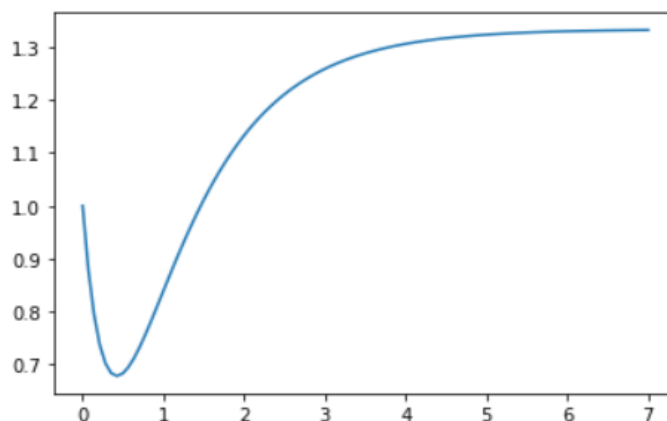
 [`<matplotlib.lines.Line2D at 0x7f3a9876a1c0>`]



**Méthode 2 :** création de la fonction de transfert à partir de la librairie **Signal**.

```
✓ [40] sys2=signal.TransferFunction(num,den)
1s      print(sys2)
        plt.figure(2)
        t,y=signal.step(sys2)
        plt.plot(t,y)
```

```
TransferFunctionContinuous(
array([1., 2., 4.]),
array([1., 4., 3.]),
dt: None
)
[<matplotlib.lines.Line2D at 0x7f3a986e7730>]
```



### Méthode 3 : Création de la fonction de transfert à partir des **zéros** et des **pôles**.

```

0 s [▶] num2,den2=signal.zpk2tf([1],[-3,-5,1+2j,1-2j],3)
      sys3=Co.tf(num2,den2)
      print(sys3)

```

$$\frac{3 s - 3}{s^4 + 6 s^3 + 4 s^2 + 10 s + 75}$$

- Vérification des pôles et des zéros à partir de la fonction de transfert obtenue

```

1 s [▶] poles=Co.pole(sys3)
      zeros=Co.zero(sys3)
      print(poles)
      print(zeros)
      Co.pzmap(sys3)

```

```

[▶] [-5.+0.j -3.+0.j  1.+2.j  1.-2.j]
     [1.+0.j]
     (array([-5.+0.j, -3.+0.j,  1.+2.j,  1.-2.j]), array([1.+0.j]))

```

Pole Zero Map

## Quelques autres applications :

- Fonction de transfert

### Tp n 3

```
✓ 0 s [▶] #définition de la fonction de transfert du système à contrôler
num=[0.01]
den=[0.005,0.06,0.1001]
g=Co.tf(num,den)
print ('g(p) =', g)
```

$$g(p) = \frac{0.01}{0.005 s^2 + 0.06 s + 0.1001}$$

- Fonction de transfert échantionnée

```
✓ 0 s [▶] #définition de la fonction de transfert échantionnée
num=[0.009201,0.005709]
den=[1,1.088,0.2369]
gz=Co.tf(num,den)
print ('gz(p) =', gz)
```

$$gz(p) = \frac{0.009201 s + 0.005709}{s^2 + 1.088 s + 0.2369}$$

- Fonction de transfert discret du régulateur PID

```
✓ 0 s [53] #définition de la fonction de transfert discrète du régulateur PID
num=[10,100,200]
den=[0,1,0]
r=Co.tf(num,den)
print ('r(s) =', r)
```

$$r(s) = \frac{10 s^2 + 100 s + 200}{s}$$

- **Fonction de transfert discrète du régulateur PID**

```
✓ [54] #définition de la fonction de transfert discrète du régulateur PID échantionnée
2s
num=[278.7,309.3,78.67]
den=[1,0,-1]
rz=Co.tf(num,den)
print ('rz(p) =', rz)

rz(p) =
278.7 s^2 + 309.3 s + 78.67
-----
s^2 - 1
```