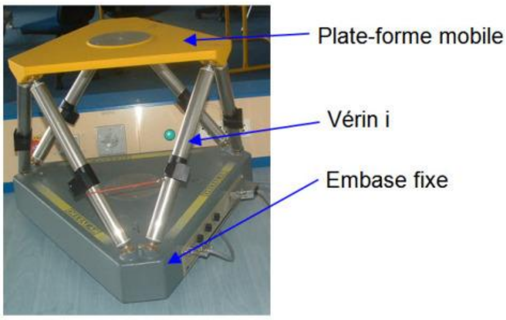
TP8
Régression linéaire

Objectifs:

- Utiliser la régression linéaire pour identifier le modèle d'une acquisition

I. importation et analyse des données:



On souhaite exploiter la mesure du déplacement d'un axe asservi de la plateforme 6 axes du laboratoire. Les données récupérées lors de l'acquisition se trouve dans le fichier csv acquisition_verin_plateforme_brute. On y retrouve dans la première colonne, les valeurs de temps et dans la deuxième la position $y(t)$ du vérin.

L'objectif de ce TP est de retrouver une fonction approchée $\tilde{y}(t)$ correspondant au mieux aux positions réelles mesurées.

Après avoir ouvert le fichier et observé les données y figurant, écrire dans l'éditeur le début de code python ci-dessous permettant d'importer les données sous forme de deux listes.

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

#ouverture du fichier et remplissage des listes
f=open("acquisition_verin_plateforme_brute.csv")
f.readline() # saut de la première ligne (non prise en compte des intitulés)
t=[]; y=[]
for l in f:
    l_sep=l.split(";")
    t.append(float(l_sep[0]))
    y.append(float(l_sep[1]))
```

1) Compléter ce code afin de tracer l'évolution de la position y en fonction du temps.

Etant donné l'affichage du résultat, on peut modéliser le comportement du vérin comme un système du premier ordre dont l'évolution de la position peut se mettre sous la forme d'une fonction approchée $\tilde{y}(t) = A.(1 - e^{-B.t})$.

On souhaite identifier les constantes A et B afin de déterminer complètement le modèle permettant de décrire le comportement du vérin.
La loi n'est pas linéaire, on ne peut donc pas faire directement une régression linéaire. En revanche, en passant par la dérivée, il sera possible de le faire grâce à un changement de variable simple.

2) Déterminer la fonction $\tilde{v}(t) = \frac{d\tilde{y}(t)}{dt}$. En posant $\tilde{V}(t) = \ln(\tilde{v}(t))$ montrer que $\tilde{V}(t)$ peut se mettre sous la forme $\tilde{V}(t) = a.t + b$.
On exprimera a et b en fonction de A et B.

On souhaite donc à partir de la liste des valeurs expérimentales de position y déterminer la liste des valeurs expérimentales de la vitesse v.

3) En utilisant l'approximation de la dérivée par la pente d'une corde, écrire une fonction **derive**(t,y) d'arguments d'entrée deux listes de même longueur, la liste t des valeurs de temps, et la liste y des valeurs de position et retournant la liste des valeurs de vitesse.

Remarque: on fera attention au fait que la liste retournée par la fonction **derive** a une valeur de moins que les listes de départ.

4) Tracer la courbe décrite par les valeurs de v en fonction du temps. Tracer maintenant la courbe des valeurs prises par V, il serait intéressant pour cette dernière de la tracer sous la forme d'un nuage de points afin d'observer la dispersion de ceux-ci.

Remarque: Lorsque l'on pose $\tilde{V}(t) = \ln(\tilde{v}(t))$, la fonction théorique $\tilde{v}(t)$ étant positive le changement de variable ne pose pas de problème. En revanche, pour les valeurs expérimentales, les valeurs réelles prises par v pouvant être négatives, il sera plus prudent de poser $V(t) = \ln(|v(t)|)$.

II. Méthode de régression linéaire par la méthode des moindres carrés

On considère un nuage de n points de coordonnées $M_i(x_i, y_i)$. On considère que le meilleur modèle associé à ce nuage de points est une relation linéaire entre les grandeurs x et y de la forme $y(x) = a.x + b$. L'objectif est de donner la meilleure estimation des coefficients $a\ et\ b$.
La méthode consiste à calculer la somme des carrés des distances prises verticalement entre les points M_i et la droite d'équation $y(x) = a.x + b$. Cette somme qui est une fonction de $a\ et\ b$ s'exprime donc de la manière suivante:

$$\sum_{1}^{n} (y_i - (a.x_i + b))^2 = f(a, b)$$

La méthode des moindre carrés consiste à minimiser cette somme, on a ainsi les deux relations suivantes:

$$\begin{cases} \dfrac{\partial f(a,b)}{\partial a} = 0 \\ \dfrac{\partial f(a,b)}{\partial b} = 0 \end{cases}$$

1) A partir des deux relations précédentes, démontrer que:

$$a = \frac{n(\sum_1^n x_i y_i) - (\sum_1^n x_i)(\sum_1^n y_i)}{n(\sum_1^n {x_i}^2) - (\sum_1^n x_i)^2}$$

$$b = \frac{(\sum_1^n {x_i}^2)(\sum_1^n y_i) - (\sum_1^n x_i)\,(\sum_1^n x_i y_i)}{n(\sum_1^n {x_i}^2) - (\sum_1^n x_i)^2}$$

2) Ecrire une fonction python RegLin(xi,yi) d'arguments d'entrées deux listes xi et yi et retournant le couple de valeurs $a\ et\ b$ définies précédemment.

3) Utiliser cette fonction afin de trouver les constantes a et b permettant de définir la forme approchée recherchée $\tilde{V}(t) = a.t + b$.

4) Tracer sur une même figure et de deux couleurs différentes, le nuage de points de l'expérimentation ($V(t)$) ainsi que la droite de régression linéaire de la forme $\tilde{V}(t) = a.t + b$.

5) Le coefficient de corrélation permet d'apprécier si le nuage de points est concentré autour de la droite ou non. Il varie de -1 à 1. Lorsqu'il est proche de 1 en valeur absolue, c'est qu'il y a une bonne corrélation, lorsqu'il est proche de 0, c'est que le nuage est très dispersé. Il se calcule de la façon suivante:

$$r = \frac{n(\sum_1^n x_i y_i) - (\sum_1^n x_i)(\sum_1^n y_i)}{\sqrt{(n(\sum_1^n {x_i}^2) - (\sum_1^n x_i)^2)(n(\sum_1^n {y_i}^2) - (\sum_1^n y_i)^2)}}$$

Ecrire une fonction python Corr(xi,yi) d'arguments d'entrées deux listes xi et yi et retournant le coefficient de corrélation.
Tester ce coefficient pour le nuage de points correspondant à notre expérience.

6) On peut retrouver le même résultat grâce à la bibliothèque scipy. Ecrire les lignes suivantes et comparer les valeurs de a, b et r obtenues précédemment.

```
from scipy.stats import linregress
(a1,b1,r1,_,_)=linregress(t[1:],V)
print(a1,b1,r1)
```

7) A partir des valeur obtenues pour a et b, en déduire les valeurs de A et B correspondant à notre modèle initial $\tilde{y}(t) = A.(1 - e^{-B.t})$.

8) Tracer sur un même graphe et de couleurs différentes, le nuage de points des valeurs expérimentales de $y(t)$ et la courbe du modèle approchée $\tilde{y}(t) = A.(1 - e^{-B.t})$.

III. Pour aller plus loin.

On souhaite appliquer la même méthode avec un signal de mauvaise qualité et donc beaucoup plus bruité afin de tester les limites de son efficacité. Ce signal part du même type d'expérience et est censé aboutir au même modèle.
Reprendre le code précédent et l'appliquer aux données contenues dans le fichier csv "acquisition_verin_plateforme_tres_bruitée".
Remarque: il n'y a qu'une ligne à changer!

Conclure sur les limites de cette méthode.