TP 04

Utilisation de modules de bibliothèques

Savoirs et compétences :

- *Th. 3 : Utilisation de modules, de bibliothèques.*

Proposition de corrigé

Activité 1 – Mise en situation et généralité

Activité 2 – Tracé de fonctions simples

Exercice 1 – (PLT-004)

En préambule :

```
import matplotlib.pyplot as plt
from numpy import exp, linspace, pi, sin, cos
from math import floor
```

Question 1

x est une liste python, donc de type `list`.

Question 2

Python représente une fonction comme une ligne brisée. On indique les coordonnées des extrémités des segments en passant en argument la liste des abscisses et celle des ordonnées à la fonction plot.

Question 3

Le tracé s'arrete au point d'abscisse 9,5. C'est bien le dernier élément de x.

Question 4

```
def ex_sin(nom\_de\_fichier):
    """Trace la courbe du sinus sur [0,10] et l'enregistre dans nom\_de\_fichier"""
    x = linspace(0,10,200)
    y = [sin(t) for t in x]

    plt.clf()
    plt.plot(x,y,label='sin(x)')
    plt.xlabel('x')
    plt.legend(loc=0)
    plt.title('Tracé du sinus sur [0,10]')
    plt.savefig(nom\_de\_fichier)
```

Question 5

```
def transitoire(A,nom\_de\_fichier):
    """Trace les graphes de t->A(1-exp(t/tau)) pour tau = 2,4,6,8 sur [0,10].
       Les sauvegarde dans nom\_de\_fichier."""
    x = linspace(0,10,200)
    tau = [0.5,1,2,4,8]
```

```
    style = ['g-','b-','r-','b--','r--']
    plt.clf()
    for k in range(5):
       y = [A*(1-exp(-t/tau[k])) for t in x]
       plt.plot(x,y,style[k],label='$\\tau='+str(tau[k])+'$')
    plt.xlabel('$t$')
    plt.ylabel('$'+str(A)+'\\times\\exp(-t / \\tau)$')
    plt.title('Régime transitoire, A={}'.format(A))
    plt.axis([0,10,0,A])
    plt.legend(loc=0)
    plt.savefig(nom\_de\_fichier)
    return None
```

Activité 3 –

0.1 Introduction

Dans cette partie, on va s'appuyer• sur un support de TP qui est utilisé aux oraux de concours : le système CoMax présenté sur la figure 1. Ce système est une adaptation pédagogique de la solution industrielle ZE de SAPELEM. Le principe de fonctionnement de ces deux systèmes repose sur l'utilisation d'un système de levage motorisé, associé à une poignée, équipée d'un capteur d'effort (voir Figure 1).

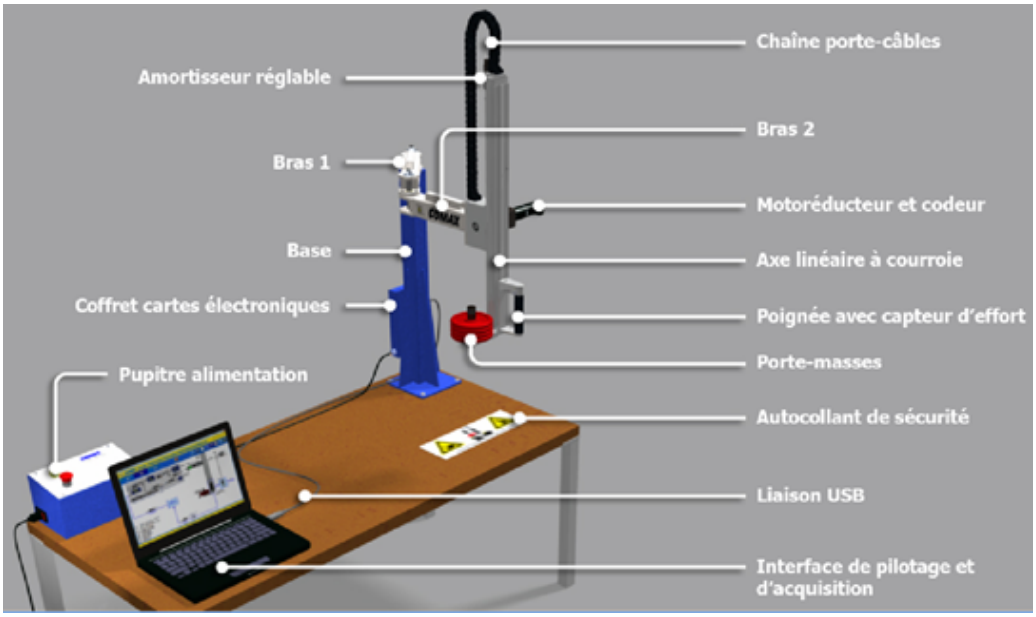

FIGURE 1: Présentation du système Comax

0.2 Extraction des données expérimentales

Nous allons dans cette partie utiliser les résultats d'expérimentions sur le robot CoMax. Ces résultats sont fournis dans le fichier *CoMax.txt*. Ils correspondent à un fichier brut avec les points de mesures expérimentales.

Question 6 *Après avoir observé la structure du fichier CoMax.txt, proposer des instructions permettant d'extraire sous deux listes distinctes* `temps` *et* `q_exp` *les instants de prise d'échantillonnage et les positions codeur correspondantes en tops (nombre de points). Ces deux listes seront converties en tableaux Numpy (*`import numpy as np`*), grâce à la commande* `liste=np.array(liste)`.

La position de l'axe linéaire $X(t)$ [en mm] est liée à la position du moteur q_m [en tops] renvoyée par le codeur selon la formule suivante : $X = \dfrac{3,41.q_m}{1000}$

Question 7 *Générer alors un tableau des positions verticales de l'axe nommé* `X_exp`*, qui a une condition*

initiale nulle sur la position.

Question 8 *Tracer l'évolution des positions mesurées expérimentales en fonction du temps, avec des croix bleues (+). On légendera correctement les axes, et on indiquera une légende du type : "points mesurés"*

0.3 Étude des performances attendues du système

Dans un second temps, nous allons modéliser le comportement attendu système. La modélisation du système est faite en amont du système réel, lors de la phase de conception, mais elle est importante pour comprendre un système réel afin de proposer des modifications affectant les performances.

Les réponses attendues en vitesse V(t) et en position X(t) de l'axe linéaire sont représentées en Figure 2 . Les caractéristiques de la loi en vitesse sont les suivantes :

- l'instant de début de mouvement : $t_0 = 0s$;
- la position initiale et la vitesse initiale : $X(0) = X_0 = 0m$ et $V(0) = 0m/s$;
- l'accélération : $A_{cmax} = 2,83m/s^2$;
- la vitesse maximale : $V_{max} = 0,68m/s$.

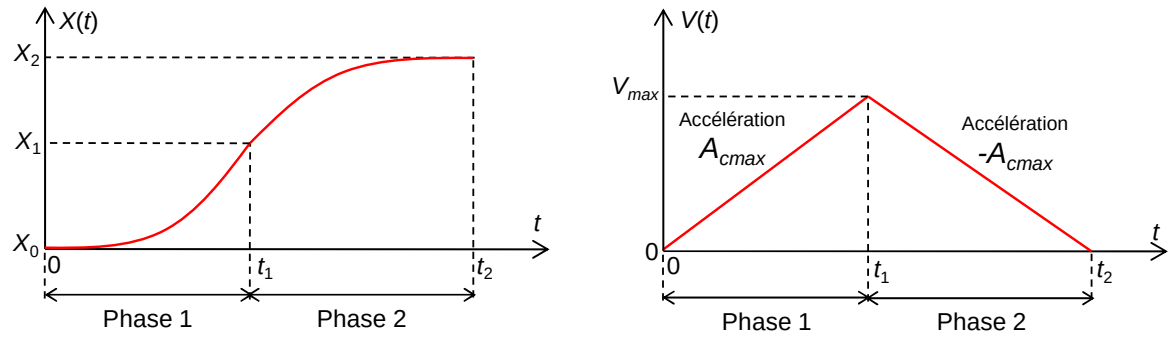

FIGURE 2: Lois d'évolution en position (à gauche) et en vitesse (à droite) du CoMax

On peut montrer par intégration successives, en tenant compte des conditions initiales que :

Phase	**1**	**2**
instant t	$0 \leq t < t_1$	$t_1 \leq t \leq t_2$
accélération $A(t)$	A_{cmax}	$-A_{cmax}$
vitesse $V(t)$	$A_{cmax}.t$	$V_{max} - A_{cmax}.(t - t_1)$
position $X(t)$	$\frac{1}{2}.A_{cmax}.t^2$	$X_1 + V_{max}.(t - t_1) - \frac{1}{2}.A_{cmax}.(t - t_1)^2$

On notera que : $X_1 = X(t_1)$, à l'issue de la phase 1 et $X_2 = X(t_2)$, à l'issue de la phase 2.

Question 9 *Trouver les expressions littérales de t_1, t_2, X_1 et de X_2 en fonction de A_{cmax} et de V_{max}.*

Question 10 *Concevoir deux fonctions* `Loi_Vitesse` *et* `Loi_position` *prenant en argument un instant t et permettant de retourner la vitesse, respectivement la position, à cet instant. (A noter que l'on pourra introduire des variables globales, comme t_1, t_2, etc.)*

On rappelle que la commande `y = np.zeros(n)` permet de générer un tableau de taille $n \times 1$ contenant des 0.

Question 11 *Construire deux tableaux* `X_th` *et* `V_th` *où sont stockées les positions théoriques commandées, respectivement vitesses théoriques aux instants définis dans le tableau* `temps`*. Superposer la courbe d'évolution de la position théorique sur les points expérimentaux, obtenus précédemment (tracé en vert, ligne continue, avec légende explicite). Sur une nouvelle figure tracer en vert, trait continu, l'évolution de la vitesse théorique*

.

0.4 Quantification et analyse des écarts

Nous allons quantifier l'écart de performance entre l'exigence du cahier des charges et le système réel. Pour cela, nous allons calculer les écarts et utiliser des outils de statistique pour quantifier ces écarts.

On définit l'écart relatif en % $\delta_{\%}$ entre une valeur théorique x_{th} et une valeur expérimentale x_{exp} : $\delta_{\%} = \left| \frac{x_{exp} - x_{th}}{x_{th}} \right|$

Question 12 *Concevoir une fonction* `Calcul_ecarts` *prenant en arguments deux tableaux à une dimension et retournant un tableau* `Delta_X` *de même dimension où sont stockés les écarts relatifs entre chacune des valeurs des deux tableaux spécifiés en arguments d'entrée.*

Question 13 *Tracer un histogramme montrant l'évolution des écarts relatifs en position en fonction du numéro de la mesure (on utilisera* `plt.bar` *et* `plt.subplot`*). On n'évaluera pas l'écart relatif sur la première valeur (nulle).*

On rappelle ici quelques notions de statistiques :

- La médiane d'un ensemble de valeurs (échantillon, population, distribution de probabilités) est une valeur qui partage la série en deux parties de même effectif.
- L'écart type est défini comme la moyenne quadratique des écarts par rapport à la moyenne. Il a la même dimension que la variable statistique étudiée.

Question 14 *Écrire une fonction* `Calculs_stats` *permettant, à partir d'un tableau* `T` *passé en argument, de retourner un tuple de 3 valeurs : moyenne, médiane et écart type. Indication : On pourra utiliser les fonctions de la bibliothèque numpy (*`np.sum(T)`*, pour faire la somme de tous les éléments du tableau* `T`*,* `np.sort(T)`*, pour trier le tableau* `T` *dans l'ordre croissant. Appliquer le résultat au tableau* `Delta_X`*.*

Question 15 *Comparer les résultats en utilisant les fonctions suivantes :* `np.mean`*, pour la moyenne ;* `np.median`*, pour la médiane et* `np.std`*, pour l'écart type*