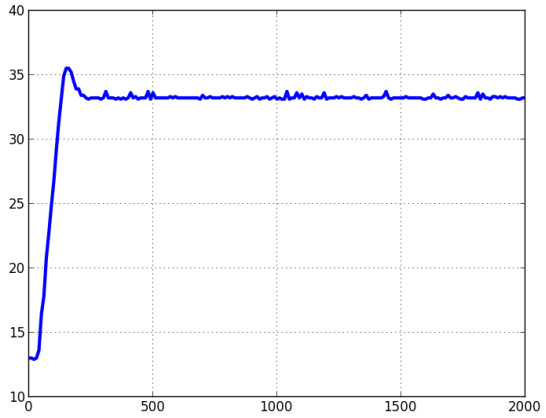
CONCOURS BLANC – 1H30

IDENTIFICATION DU COMPORTEMENT D'UN SYSTÈME ASSERVI

Objectif

L'objectif de ces travaux est de déterminer automatiquement les caractéristiques d'un système asservi en le modélisant par une fonction de transfert du premier ou du second ordre. Ainsi, à partir d'un relevé expérimental, les programmes réalisés doivent permettre de déterminer les caractéristiques des systèmes.

On donne ci-contre un exemple de mesures brutes pouvant être acquises.



Exemple de mesure pouvant être modélisée par un système du second ordre

Dans le cadre de sujet, les systèmes seront sollicités par des échelons d'amplitude E_0. On suppose que l'amplitude du bruit est inférieur à 0,1 (l'unité dépendant du signal mesuré).

1 Préambule

Les logiciels d'acquisitions utilisés dans le cadre expérimental permettent d'obtenir un fichier texte encodé en ASCII. Ce fichier est composé de deux colonnes. La première contient les temps de mesures en secondes. La seconde contient les informations mesurées.

Pour un relevé courant, la fréquence d'échantillonnage de la mesure est de 1 kHz. La mesure est comprise entre -999 et 999 avec une précision au millième. (Par exemple, pour une mesure de déplacement en millimètres, on fait l'hypothèse que la mesure sera comprise entre -999 et 999 mm avec une précision de 1 micromètre.)

Question 1

Quel type de variable est présent dans le fichier texte ? Quel type de variable proposeriez vous pour stocker un élément de la première colonne ? un élément de la seconde colonne ? une colonne toute entière ?

Question 2

Quelle sera la taille d'un fichier de mesure courant dans le cas le plus défavorable (mesure de 2 secondes) ?

Question 3

On donne la séquence d'instruction suivante. Expliquer l'objectif de chacune des lignes.

python

```
1  nom_fichier="fichier_mesure.dat"
2  fid  = open(nom_fichier,'r')
3  donnees_fichier=fid. readlines ()
4  fid . close ()
```

Une fonction appelée traitementDonnees permet de retourner :

– les temps de mesures (que l'on stocke dans un tableau appelé temps) ;
– les mesures elles-mêmes (que l'on stocke dans un tableau appelé mesures).

On rappelle que dans le cas d'un système du premier ordre, la tangente à l'origine est non nulle. Dans le cas d'un système du second ordre la tangente à l'origine est nulle.

On donne la fonction ordre qui permet de déterminer si on peut identifier le comportement du système à un premier ou à un second ordre.

```python
1  def ordre(temps,mesures):
2      j=0
3      for i in range(len(mesures)-1):
4          j=i
5          if abs(mesures[i]-mesures[i+1])>0,1:
6              break
7      pente = (abs(mesures[j+1]-mesures[j]))/(temps[j+1]-temps[j])
8      if pente <0,1:
9          return 2
10     else :
11         return 1
```

Question 4

Expliquer l'objectif de la boucle for (lignes 3 à 6) ?

Question 5

Expliquer l'objectif des lignes 7 à 11 ? À quoi correspond le test de la ligne 8 ?

Pour la suite, on dispose de la fonction calculDebut qui permet de retourner l'index à partir duquel la mesure commence à varier. Elle prend comme argument le tableau de mesures.

2 Identification d'un système d'ordre 1

On rappelle qu'un système d'ordre 1 est de la forme $H(p)=\dfrac{K}{1+\tau p}$. Le programme doit d'une part permettre de déterminer le gain K du système et d'autre part permettre de déterminer la constante de temps 3 suivant 3 méthodes :

- l'utilisation de la tangente à l'origine ;
- les 63% de la valeur finale ;
- les 95% de la valeur finale.

On rappelle que théoriquement, K est déterminé en régime permanent. On a alors :

$$K=\frac{s(\infty)-s(0)}{e(\infty)-e(0)}$$

En notant T la durée des essais réalisés, on suppose que le régime permanent est atteint au moins avant 80% T.

Question 6

Réaliser une fonction simple appelée calculGain prenant comme argument le tableau mesures et l'amplitude E0 de l'échelon d'entrée et retournant la valeur du gain statique K.

Question 7

Pour une plus grande fiabilité des résultats pour le calcul de K, on désire utiliser une moyenne sur les 20 derniers pour cents d'une mesure. Quel en est l'intérêt ? Implémenter une fonction nommée calculValeurFinale permettant de déterminer la valeur finale. Implémenter alors une fonction nommée calculGainMoyen permettant de déterminer K en tenant compte de ces nouvelles contraintes.

Question 8

En prenant comme critère le fait que le signal a atteint 63% de sa valeur finale au bout d'un temps τ, compléter la fonction calculTau permettant de déterminer la constante de temps du système en utilisant un algorithme de recherche dichotomique. On rappelle l'existence de la fonction calculDebut.

```python
def calculTau(temps,mesures,E0):
    i0 = calculDebut(mesures)                # Index du depart de la mesure
    ifin = len(mesures)                      # Index de la derniere mesures
    mes0 = mesures[i0]                       # Valeur initiale
    mesf = calculValeurFinale (mesures)      # Calcul de la valeur finale
    mes_63 = 0,63*(mesf-mes0) +mes0
    ind_g = i0
    ind_d = ifin
    mes_g=mesures[ind_g]
    mes_d=mesures[ind_d]
    while
        ind_m = int((ind_g+ind_d)/2)
        mes_m =mesures[ind_m]
        if


        else :


    return
```

Question 9

Que pouvez-vous dire de la précision de la valeur de la constante de temps ? Serait-il possible de trouver une meilleure approximation ? Si oui, expliquer succinctement votre démarche.

Question 10

Quelle est la complexité de cet algorithme ? Serait-il possible d'estimer le nombre d'itération de la boucle while ? Quelle serait la complexité d'un algorithme de recherche naïf ? Quel est l'algorithme le plus efficace temporellement ?

Question 11

Réaliser la fonction identificationOrdre1 permettant de calculer le gain et la constante de temps. Cette fonction prendra comme argument les tableaux temps et mesures.

3 Identification d'un système d'ordre 2

On rappelle qu'un système d'ordre 2 est de la forme $H(p)=\dfrac{K}{1+\dfrac{2\xi}{\omega_0}p+\dfrac{1}{\omega_0^2}p^2}$. Le programme doit permettre de déterminer le gain K du système, le coefficient d'amortissement ξ et la pulsation propre ω_0. Le gain peut être calculé avec la même méthode que dans la partie précédente. On rappelle que théoriquement :

- ξ est tel que $D_{1\%}=e^{-\frac{\pi\xi}{\sqrt{1-\xi^2}}}$ avec $D_{1\%}$ le premier dépassement pour cent.
- ω_0 est tel que : $T_p=\dfrac{2\pi}{\omega_0\sqrt{1-\xi^2}}$ avec T_p la pseudo période.

On s'intéresse ici uniquement à la détermination de premier dépassement pour cent et de la pseudo période. On rappelle que le premier dépassement peut être défini par le rapport $\dfrac{s_{max}-s(\infty)}{s(\infty)-s(0)}$.

La fonction calculAmortissement prend la valeur du premier dépassement pour cent et retourne le coefficient d'amortissement ξ. La fonction calculPulsation prend comme argument T_p et ξ et retourne la pulsation propre du système.

On rappelle que la pseudo période peut être calculée en multipliant par 2 le temps auquel survient le premier dépassement.

Question 12

Donner une méthode permettant de déterminer automatiquement le premier dépassement à partir d'une mesure.

Question 13

Réaliser la fonction premierDepassement permettant de savoir quand a lieu le temps de premier dépassement et quelle en est sa valeur.

Question 14

On souhaite savoir à quel moment a lieu l'intersection entre la mesure et l'asymptote horizontale lors de la première montée. Donner une méthode permettant de déterminer ce temps de montée.

La fonction pseudoPeriode permet de calculer la pseudo période à partir du fichier de mesures.

Question 15

Réaliser la fonction identificationOrdre2 permettant de calculer le gain la pulsation propre et le coefficient d'amortissement. Cette fonction prendra comme argument les tableaux temps et mesures ainsi que E_0 (amplitude de l'échelon d'entrée).

4 Synthèse

Question 16

Réaliser la fonction identification permettant de déterminer dans un premier temps si la mesure réalisée est d'ordre 1 ou d'ordre 2. Dans un second temps, cette fonction devra calculer les paramètres caractéristiques des fonctions de transfert. Cette fonction prendra comme argument une chaîne de caractère correspondant au nom du fichier de mesures. Elle retournera les caractéristiques du système.

Question 17

On désire avoir recours à la dérivée du signal mesuré. Réaliser la fonction deriveMesure prenant comme argument les tableaux temps et mesures. Cette fonction renvoie une liste de couples [temps,derivee].

Question 18

On désire avoir recours à l'intégrale du signal mesuré. Réaliser la fonction integreMesure prenant comme argument les tableaux temps et mesures. Cette fonction renvoie une liste de couples [temps,integrale].